jamk.fi

Nagios-monitoroinnin asennus Contriboard-testiympäristöön

Eetu Kyckling

Opinnäytetyö
Toukokuu 2016
Tekniikan ja liikenteen ala
Insinööri (AMK), tietotekniikan koulutusohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Kyckling, Eetu	Opinnäytetyö, AMK	23.05.2016
	Sivumäärä 66	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi
Nagios-monitoroinnin asennus Contriboard-testiympäristöön

Koulutusohjelma
Tietotekniikan koulutusohjelma

Työn ohjaaja(t)
Mika Rantonen, Antti Häkkinen

Toimeksiantaja(t)
Jarmo Viinikanoja, N4S@JAMK

Tiivistelmä

Työn tarkoituksena oli asentaa Nagios-monitorointisovellus N4S@JAMK:n Contriboard-palvelun testiympäristöön ja dokumentoida asennus niin, että Nagios voidaan niillä ohjeilla asentaa tuotantoympäristöön.

Contriboard-palvelua suoritetaan Amazonin pilvipalvelun Linux-palvelimilla, joten asennus on tehtävä SSH-yhteyden avulla pelkillä terminaalikomennoilla.

Tavoitteena oli saada Contriboard-palvelua ylläpitävälle taholle reaaliaikainen näkymä, jossa on myös mahdollisuus tallentaa lokitietoja kultakin palvelimelta, jotta voidaan vikatilanteessa selvittää, mikä on aiheuttanut ongelman.

Tämän lisäksi opinnäytetyön tavoitteena oli mahdollisuuksien mukaan automatisoida uusien kohteiden lisäämistä valvonnan piiriin esimerkiksi Ansible-työkalulla tai eri skripteillä. Myös Dockerin käyttö Nagioksen asennuksessa tuli ottaa huomioon.

Opinnäytetyön tuloksena saatiin tehtyä yksityiskohtaiset ohjeet Nagios-monitorointisovelluksen asentamiseen Linux-ympäristöön. Tämän lisäksi opinnäytetyössä tehtiin skriptit, jotka yksinkertaistavat uuden palvelimen lisäämistä valvonnan piiriin. Työssä on lisäksi huomioitu Dockerin hyödyntäminen Nagioksen käyttöönotossa testaamalla Nagioksen suorittamista kontista sekä konfiguraatioiden siirtämistä valmiista ympäristöstä konttiin sisälle.

Avainsanat (asiasanat)
Contriboard, monitorointi, Linux, Nagios, palvelin, testiympäristö, Docker, skripti

Muut tiedot

jamk.fi

Description

Author(s)	Type of publication	Date
Kyckling, Eetu	Bachelor's thesis	23.05.2016
		Language of publication: Finnish
	Number of pages 66	Permission for web publication: x

Title of publication

Installing Nagios monitoring software to Contriboard test environment

Degree programme

Information Technology

Tutor(s)

Mika Rantonen, Antti Häkkinen

Assigned by

Jarmo Viinikanoja, N4S@JAMK

Abstract

The objective of this bachelor's thesis was to install Nagios monitoring software into N4S@JAMK's Contriboard services test environment and to document the installation for the future installation of Nagios into production environment of Contriboard.

Contriboard is run from Amazon's cloud computing service and it uses Linux-servers, which means that the installation must be implemented with SSH connection using only terminal commands.

The main goal was to have a real-time monitoring view for Contriboard maintenance, where it is possible to save log files from each of the servers, which helps to resolve the cause of a problem in a fault situation.

The thesis project also needed to find out if it is possible to automate the adding of new hosts under Nagios-monitoring by using Ansible or scripts. The possibility to make use of Docker when installing Nagios had to be taken into consideration.

The results of this thesis gives detailed instructions on how to install Nagios monitoring software into any environment with Linux-servers. It is also documented how to use scripts to automate the installation of a new server, which simplifies the process what needs to be configured. Thesis also takes in consideration how Docker could be introduced in installing Nagios and running it from a container.

Keywords/tags (subjects)

Contriboard, monitoring, Linux, Nagios, server, test environment, Docker, script

Miscellaneous

Sisältö

Kuviot

Taulukot

Lyhenneluettelo

AWS	Amazon Web Service
CGI	Common Gateway Interface
CLI	Command Line Interface
CPAN	Comprehensive Perl Archive Network
EC2	Elastic Compute Cloud
JAMK	Jyväskylän ammattikorkeakoulu
LAMP	Linux, Apache, MySQL/MariaDB, PHP/PERL/Python
MRTG	Multi Router Traffic Grapher
N4S	Need for Speed
NRPE	Nagios Remote Plugin Executor
OSI	Open Systems Interconnect
PERL	Practical Extraction and Report Language
RRD	Round-robin database
SSH	Secure Shell
SSL	Secure Sockets Layer
TEKES	Teknologian ja innovaatioiden kehittämiskeskus
WWW	World Wide Web

1 Opinnäytetyön lähtökohdat

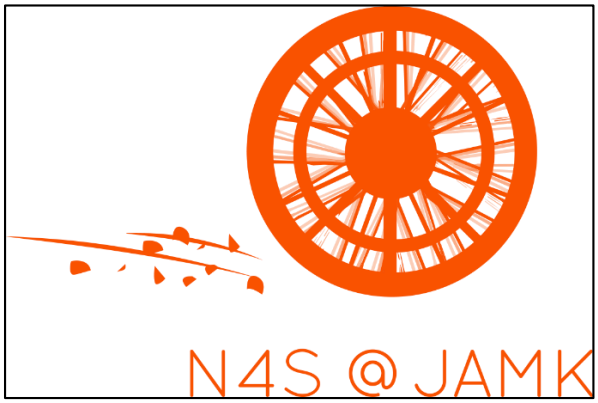
1.1 Tehtävänanto

Opinnäytetyön tavoitteena oli saada dokumentoitua Nagios-monitorointisovelluksen asennus testiympäristöön niin, että niillä ohjeilla se voidaan asentaa Contriboard-tuotantoympäristöön. Tehtävänantoon kuului konfiguroida Nagios niin, että sitä voidaan seurata reaaliajassa ja tarvittaessa sen avulla saadaan lokitietoa siitä, mitä on tapahtunut palvelussa ongelmatilanteen tullessa ilmi. Tämän lisäksi Nagios voidaan tarvittaessa konfiguroida niin, että se lähettää sähköpostin tietyn tapahtuman sattuessa, esimerkiksi jonkin palvelimen tai palvelun vikatilanteessa.

Lisäksi työhön kuului selvittää, onko mahdollista automatisoida Nagioksen asentamista Contriboard-ympäristöön esimerkiksi joko Fabric- tai Ansible-työkaluilla tai mahdollisilla skripteillä, sekä voiko Docker-työkalua käyttää Nagioksen asennuksen helpottamiseksi.

1.2 N4S@JAMK

N4S@JAMK on Digilen yhteistyökonsortio, jonka tarkoituksena on tarjota suomalaisille ohjelmistoyrityksille menestystä reaaliaikaisten liiketoimintamallien kokeiluilla. TEKES toimii Digile-ohjelman rahoittajana. Tällä yhteistyöllä JAMK:n kanssa halutaan kehittää reaaliaikaisia ohjelmistoja, joista Contriboard on yksi. Kuviossa 1 on esitetty N4S@JAMK:n logo. (N4S-ohjelma n.d.)



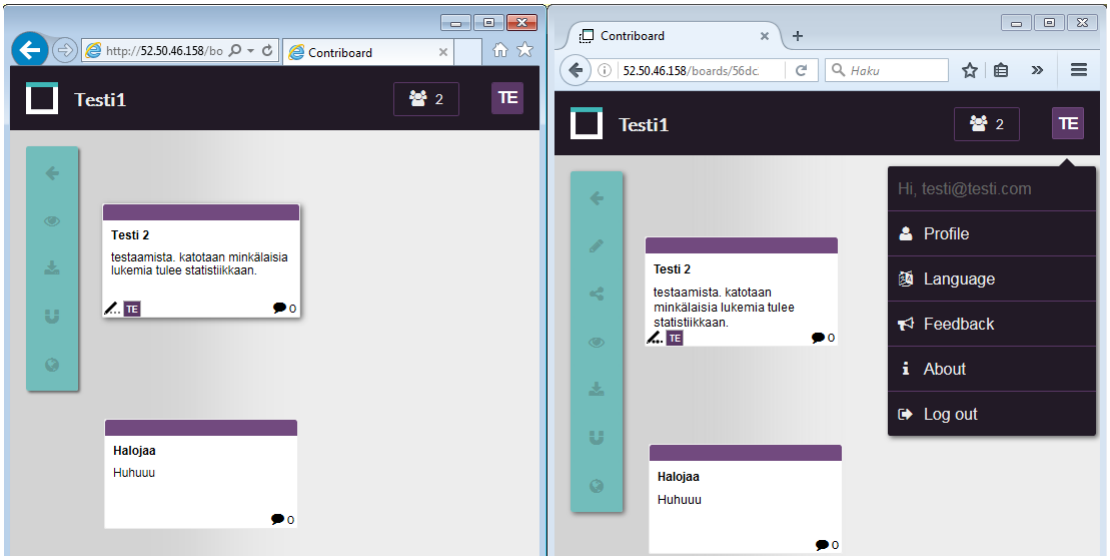
Kuvio 1. N4S@JAMK-logo (N4S@JAMK n.d.)

1.3 Contriboard

Contriboard on selaimella käytettävä, reaaliajassa useammalle käyttäjälle toimiva muistilappujärjestelmä. Siinä samanaikaisesti useampi käyttäjä pystyy esimerkiksi kokouksen aikana lisäämään omia muistilappujaan, ja muutokset näkyvät välittömästi kaikille, jotka kyseistä ”taulua” käyttävät. Contriboard-palvelua ylläpidetään Amazonin EC2-pilvipalvelussa (Elastic Compute Cloud) Linux-virtuaalipalvelimilla (Ubuntu-distribuutio). Kuviossa 2 on esitetty Contriboard-logo. (Contriboard N.d.)

Kuvio 2. Contriboard-logo (Contriboard n.d.)

Palvelun kriittisimmät osat ovat verkkoyhteyksien toiminta eri osien välillä. Yhteyksien täytyy toimia mahdollisimman pienillä viiveillä, jotta muistilaput toimivat reaaliaikaisesti kaikilla käyttäjillä. Siksi on erityisen tärkeää valvoa yhteyksien toimintaa verkkoa reitittävältä HAProxy-palvelimelta Contriboardin eri osiin kuin myös Contriboardin tietokantoja ylläpitäviin portteihin. Kuviossa 3 on esitetty Contriboardin samanaikaista toimintaa kahdelta eri selaimelta, kahdella eri käyttäjällä.



Kuvio 3. Esimerkkikuva Contriboardin toiminnasta

2 Teoria

2.1 Verkon monitorointi

2.1.1 **Yleistä**

Verkon ja siinä olevien laitteiden monitorointi voidaan lukea kriittisen tärkeäksi, riippuen palvelusta, jota yritys tai yhteisö tarjoaa käyttäjilleen. Monitoroinnilla voidaan saada viiveettä vikatilanteista hälytykset ylläpidosta vastaavalle henkilöstölle ja seurata ympäristön toimivuutta sekä verkkoliikenteen määrää. Verkkoa voidaan monitoroida joko siihen tarkoitukseen tehdyillä ohjelmistoilla tai verkkoon liitetyillä erillisillä laitteilla. Verkon monitorointia suunniteltaessa on yhtä tärkeää ottaa huomioon, mitä monitoroida ja mitä halutaan jättää pois valvonnan piiristä. (Network Monitoring Definition and Solutions 2007.)

Ensimmäinen ajatus verkon toiminnasta voi olla seuraava: jos se ei ole rikki, älä koske siihen. Palvelun tuottamisessa asiakkaille tämä ajatustapa on täysin väärä. Monitoroinnilla voidaan etukäteen ehkäistä syntyviä ongelmia ja suunnitella mahdolliset laajennukset sekä saadaan kokonaiskuva verkon toiminnasta. Näin ollen mahdolliset katkokset voidaan estää, mikä ehkäisee sopimuksessa määriteltyjen palvelutasojen rikkomista ja pystytään tuottamaan asiakkaalle parempaa ja varmatoimisempaa palvelua.

2.1.2 **Verkon monitoroinnin hyvät käytännöt**

Verkon monitoroinnissa on muutama käytäntö, joita kannattaa noudattaa. Nämä hyvät käytännöt on käyty läpi seuraavissa kappaleissa.

Ennen mahdollisten ongelmien huomaamista tai käyttäjien huomauttamista, on verkon ylläpitäjän tiedettävä, mikä on normaalia verkon toimintaa, eli toiminnasta on saatava peruskuva. Vasta tämän jälkeen pystytään asettamaan raja-arvoja toiminnalle. (Network Monitoring Best Practices n.d.)

Seuraava hyvä käytäntö on sopia toimintatapa häiriön sattuessa. Tapahtuma verkossa eskaloituu usein ongelmaksi siinä vaiheessa, jos mahdolliseen tilanteeseen ei reagoida tai oikeaa henkilöä ei saada hälytettyä tilanteeseen. Tämän takia on tärkeää sopia eskalointimatriisi, jossa määritellään missä tilanteessa ongelmaan hälytetään tietty henkilö. (Network Monitoring Best Practices n.d.)

Verkkoa ei ole kannattavaa monitoroida vain yhdeltä OSI-mallin (Open System Interconnection) kerrokselta, vaan verkkoa kannattaa valvoa sekä fyysisellä tasolla että sovellustasolla. Näin ollen monitorointisovellusta valitessa on hyvä ottaa huomioon, voidaanko sen avulla valvoa verkkoa useammalla eri OSI-mallin tasolla. (Network Monitoring Best Practices n.d.)

Myös monitoroinnissa olisi hyvä huomioida korkea saatavuus. Jos myös monitorointipalvelin sijaitsee samassa verkossa valvottavien kohteiden kanssa, eikä esimerkiksi verkkoyhteyttä ole kahdennettu, koko verkon kaatuessa tilanteesta ei saada minkäänlaista hälytystä, jos hälytykset on järjestetty esimerkiksi sähköpostiviesteillä. Tällöin aktiivisesta monitoroinnista ei ole apua. Kuten muissakin järjestelmissä yhden pisteen (single point of failure) varaan ei kannata asettaa kriittisiä palveluita. (Network Monitoring Best Practices n.d.)

Konfiguraatiohallinta kuuluu ennaltaehkäisevään verkon valvontaan. Ihmisen tekemät muutokset konfiguraatioissa sisältävät aina inhimillisen virheen mahdollisuuden, joten pienikin virhe voi aiheuttaa pahan katkoksen palveluissa. Järjestelmällisellä konfiguraatiohallinnalla voidaan ehkäistä syntyviä ongelmia. (Network Monitoring Best Practices n.d.)

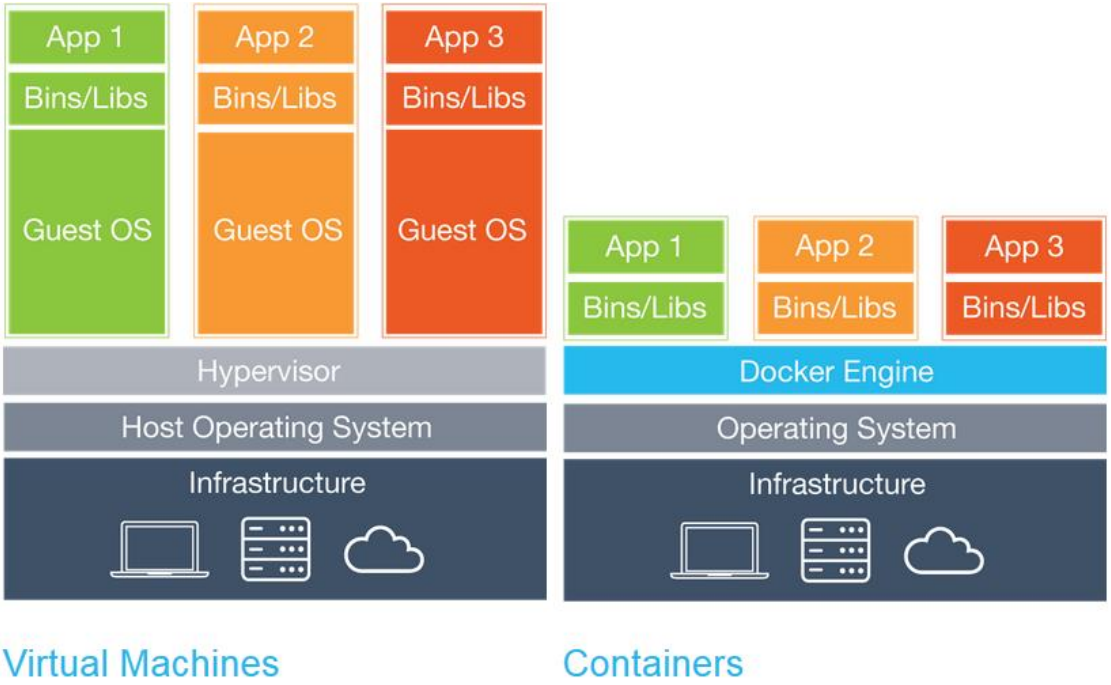
Myös kasvu on huomioitava monitorointia suunnitellessa. Verkon kasvaessa myös monitoroinnin määrä lisääntyy, joka vaikuttaa suoraan monitorointipalvelimen käyttämiin resursseihin. Lisäksi ohjelmisto voi olla lisenssipohjainen, jolloin monitorointikohteiden lisääminen voi tarkoittaa lisää kuluja. Näin ollen, jos kasvua on näkyvissä, on suositeltavaa jo monitorointia suunnitellessa ottaa huomioon nämä seikat. (Network Monitoring Best Practices n.d.)

2.2 Docker-sovelluksen teoria

2.2.1 **Yleistä Dockerista ja konteista**

Sovellusten ”kontittaminen” on kustannustehokkaampaa, kuin virtuaalipalvelimien suorittaminen. Yhdellä palvelimella voidaan suorittaa satoja, jopa tuhansia kontteja, riippuen kapasiteetista. Kontittamiseen on kehitetty avoin alusta, Docker. Sen avulla voidaan sovellusten suorittaminen tehdä erillisissä konteissa, sen sijaan että ne suoritettaisiin joko suoraan palvelimella tai sillä suoritettavalla virtuaalipalvelimella. Tämä toteutustapa mahdollistaa sovellusten nopeamman käyttöönoton ja testaamisen, ja lyhentää aikaa, joka kuluu koodin kirjoittamisen ja suorittamisen välillä. (Understand the architecture n.d.)

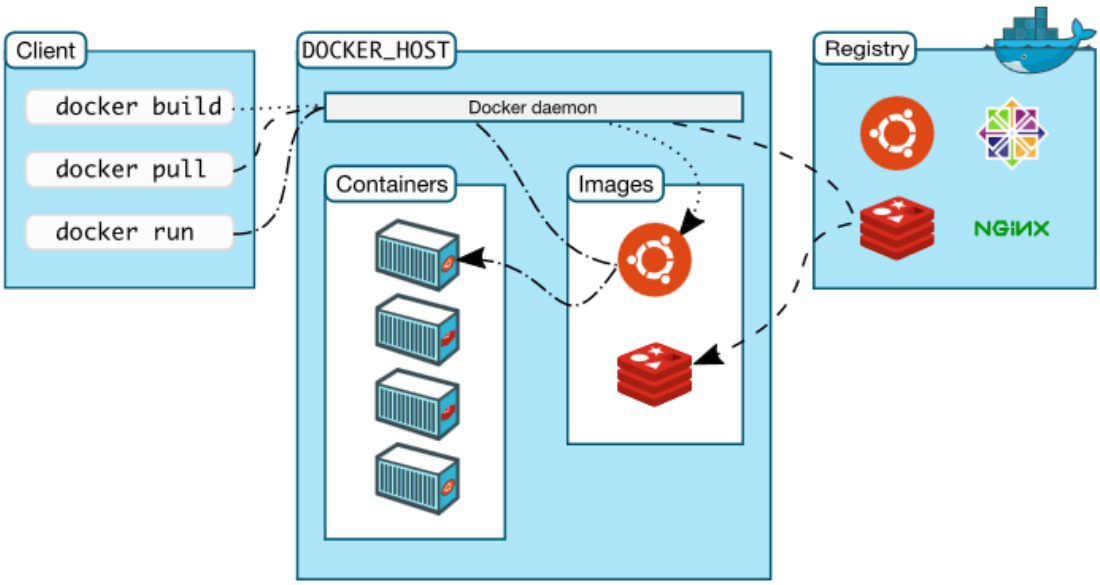
Virtuaalipalvelimien ja konttien suorittamisen erot on esitetty kuviossa 4. Suurin ero näiden järjestelmien välillä on se, että virtuaalipalvelin tarvitsee niin sanotun vieraskäyttöjärjestelmän (Guest OS), jotta palvelimella voidaan suorittaa sovellusta, kun taas kontteja hyödyntämällä sama sovellus voidaan suorittaa raudan päälle asennetulta käyttöjärjestelmältä. Tämä yksinkertaistaa ja tehostaa ympäristöä, koska raudan kapasiteettia ei käytetä ylimääräisten käyttöjärjestelmien suorittamiseen. (What is Docker? n.d.)



Kuvio 4. Virtuaalipalvelimien ja konttien ero (What is Docker? n.d.)

2.2.2 **Dockerin toimintatapa**

Docker on rakennettu toimimaan niin sanotulla asiakas/palvelin-arkkitehtuurilla, joka on esitetty kuviossa 5. Arkkitehtuuri sallii asiakassovelluksen (Client) ja palvelinsovelluksen (DOCKER_HOST/Docker daemon) asentamisen joko samalle tai eri palvelimille. Levykuvat (images) sijaitsevat rekisterissä (Registry), joista suosituin on Dockerin oma Docker Hub. Jokainen käyttäjä voi rakentaa oman rekisterinsä, johon voidaan lisätä omat levykuvat. (Understand the architecture n.d.)



Kuvio 5. Docker-arkkitehtuuri (Understand the architecture N.d.)

Konttien kokoaminen ja suorittaminen tehdään asiakassovelluksella build-, pull- ja run-komennoilla. Asiakassovellus käskyttää Docker daemonia, joka kokoaa kontin annetun käskyn mukaisesti. Jos pyydettyä levykuvaa ei ole aiemmin käytetty, se ladataan rekisterin kautta. Kokoamisen jälkeen kontti suoritetaan palvelinsovelluksella, jonka jälkeen kontin sisällä suoritettavat palvelut ovat käytettävissä DOCKER_HOST-palvelimen kautta. (Understand the architecture n.d.)

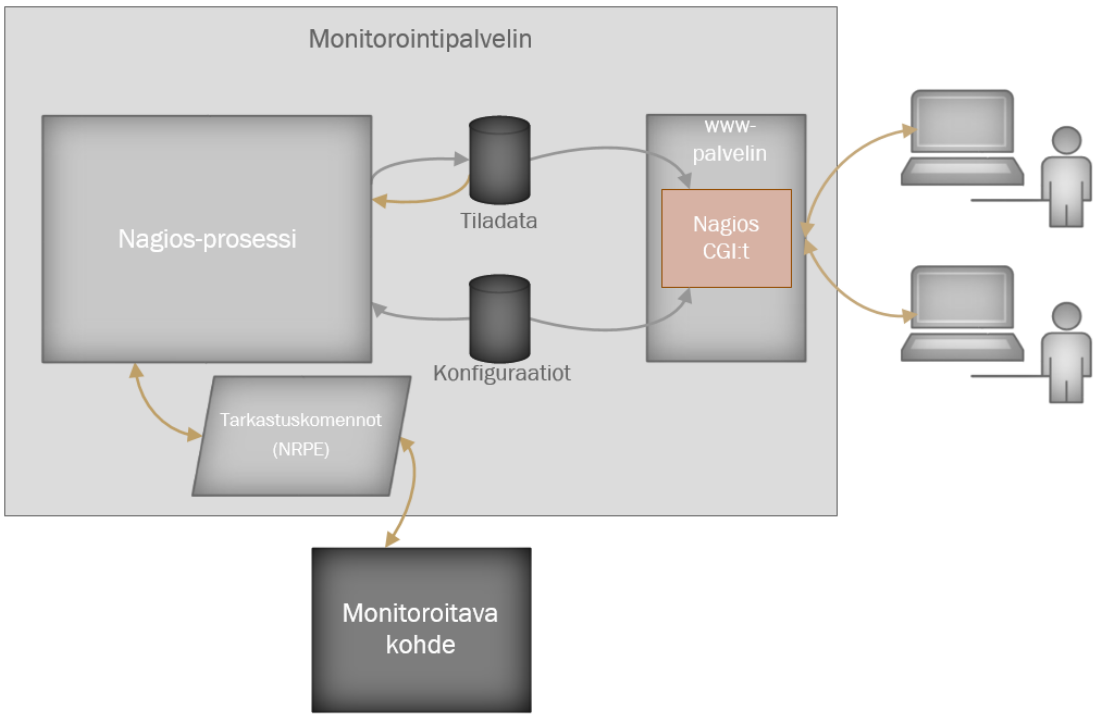
2.3 Nagios Core-teoria

2.3.1 **Lyhyt historia**

Nagios Coren (myöhemmin Nagios) kehitystyön on aloittanut Ethan Galstad vuonna 1996 yksinkertaisella MS-DOS-sovelluksella, jolla pystyi ”pingaamaan” Novell Netware-palvelimia. Vuonna 1999 Nagios on otettu Open Source projektiksi työnimellä NetSaint, mutta tavaramerkkiongelmista johtuen nimi vaihdettiin vuonna 2002. Vuodesta 2006 eteenpäin Nagiosta on palkittu vuosittain parhaana monitorointiohjelmistona. Vuonna 2009 nimi vaihtui Nagios Coreksi ja Nagios-projektista irtaantui toinen suosittu monitorointiohjelmisto Icinga. Vuonna 2013 julkaistiin Nagioksen versio 4, jota tässäkin opinnäytetyössä on käytetty. Uusin versio, Nagios XI 5 on julkaistu vuonna 2015. (About Nagios n.d.; Nagios Core Documentation n.d.)

2.3.2 **Perusperiaate**

Nagiosta suoritetaan ympäristöön asennetulta, mielellään erilliseltä palvelimelta. Tällä palvelimella suoritetaan varsinaista prosessia, säilytetään tiladata ja konfiguraatiot. Lisäksi se sisältää myös WWW-rajapinnan (World Wide Web), jota loppukäyttäjät pystyvät seuraamaan. Tarkastuskomennot ajetaan NRPE-prosessin (Nagios Remote Plugin Executor) avulla monitoroitavilla kohteilla. Yksinkertainen toimintakaavio on esitetty kuviossa 6.



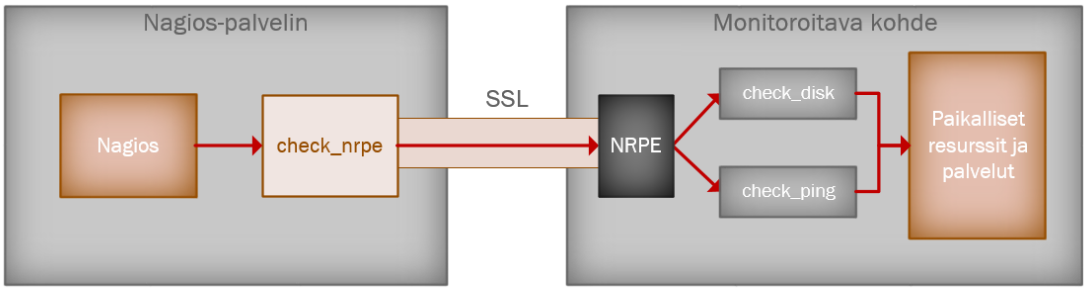
Kuvio 6. Yleiskuvaus Nagioksesta

Lyhyesti kuvailtuna Nagioksen toimintaperiaate kulkee seuraavan kaavion mukaisesti:

- Nagios-prosessi lisäosineen asennetaan palvelimelle (yleensä linux-pohjainen)
    - Kohteelle asennetaan omat työkalut
- Asennuksen yhteydessä tehdään tarvittavat konfiguraatiot, jotka tallennetaan palvelimelle
- Nagios-prosessi suorittaa tarkastuskomennot NRPE:n kautta monitoroitavalta kohteelta
- Saatu tiladata tallennetaan palvelimelle

- Erillinen www-palvelu suorittaa käyttöliittymää Nagiokselle
- Konfiguraatioilla määritetyt CGI:t ovat luettavissa www-sivuston kautta
- Loppukäyttäjä siirtyy selaimella tarkastamaan Nagioksen keräämät tiedot
- Tarvittaessa vikatilanteista saadaan hälytykset esimerkiksi sähköpostilla tai tekstiviestillä

Tarkempi kuvaus Nagioksen ja NRPE:n toiminnasta on esitetty kuviossa 7. Eli Nagioksen konfiguraatiolla määritetään suoritettavaksi check_nrpe-komento, joka tarkempien määritysten perusteella on käsky monitoroitavalle palvelimelle suorittaa paikallisesti määritetty komento. NRPE-komennot ajetaan verkkoyhteyden yli SSL-protokollalla suojattuna oletuksena portin 5666 läpi.

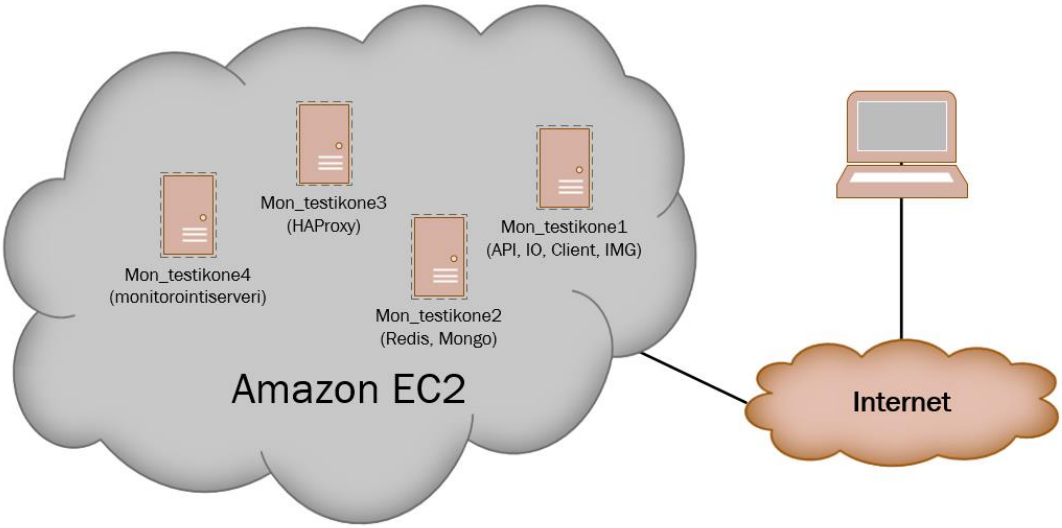


Kuvio 7. NRPE:n periaate

3 Nagioksen asentaminen ja konfigurointi

3.1 Testiympäristö

Testiympäristö muodostuu neljästä Linux-palvelimesta, joilla suoritetaan Ubuntu 14.04-käyttöjärjestelmää. Näistä neljästä koneesta yksi toimii ns. host-koneena, jolla Nagioksen pääprosessia suoritetaan, ja kolme muuta konetta suorittavat Contriboardin eri osia. Kuviossa 8 on esitetty testiympäristö ja taulukossa 1 käytössä ollut IP-osoitteistus. Sisäverkon IP-osoitteilla on merkitystä vain Contriboardin osien toiminnassa.



Kuvio 8. Opinnäytetyössä käytetty testiympäristö

Taulukko 1. Testiympäristössä käytössä ollut IPv4-osoitteistus

Virtuaalikoneen nimi	Rooli	Sisäinen IPv4-osoite	Julkinen IPv4-osoite
mon_testikone1	Contriboard-rajapinta	172.31.175.64	Vaihtuva
mon_testikone2	Contriboard-tietokanta	172.31.175.63	Vaihtuva
mon_testikone3	HAProxy	172.31.175.65	Vaihtuva
mon_testikone4	Nagios-monitorointipalvelin	172.31.175.59	Vaihtuva

Kullakin virtuaalipalvelimella on julkinen IP-osoite, jonka avulla palvelimelle saa SSH-yhteyden internetin puolelta. Ne jaetaan Amazonin läntisen Euroopan ensimmäisen alueen (EU-WEST-1) julkisen IP:n alueilta. (AWS IP Address Ranges 2015.)

3.2 Alkuvalmistelut

3.2.1 **Root-käyttäjä**

Kaikki komennot palvelimilla on syytä ajaa root-käyttäjänä. Näin ollen jokaista komentoa varten ei tarvitse erikseen siirtyä käyttämällä komentoa sudo. Root käyttäjäksi siirrytään komennolla

```
sudo su
```

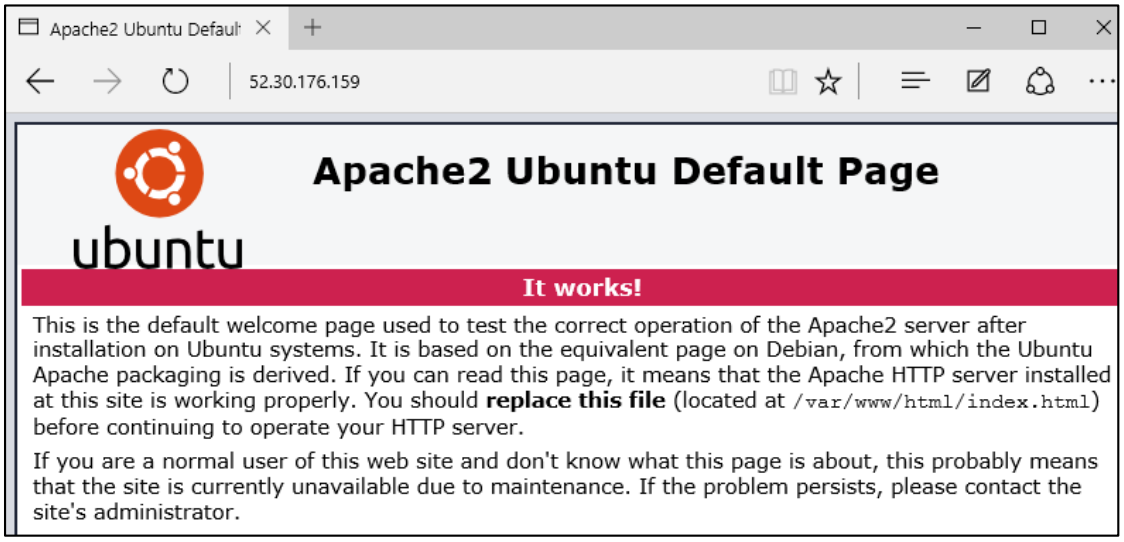
3.2.2 **LAMP-asennus**

Nagios tarvitsee toimiakseen täydellisen LAMP-asennuksen (Linux-Apache-MySQL/MariaDB, PHP/PERL/Python) palvelimella. LAMP on kokoelma avoimen lähdekoodin ohjelmia, jotka mahdollistavat WWW-palvelimen toiminnan. Nagioksen suorittamiseen palvelimella tarvitaan jokaista näistä osista. Ubuntu-palvelimen asennuspaketissa nämä sovellukset ovat yleensä jo esiasennettuna, mutta seuraavissa kappaleissa on käyty läpi näiden asennukset.

Ensimmäinen LAMP-osa on itsessään Linux-palvelin, joka ei vaadi toimenpiteitä. Seuraavana on vuorossa Apache, josta uusin versio on Apache2. Se asennetaan komennolla

```
apt-get install apache2
```

Asennuksen suoriuduttua loppuun on WWW-palvelu käynnistynyt ja toiminta voidaan testata siirtymällä internetselaimella palvelimen IP-osoitteeseen, joka on esitetty kuviossa 9.



Kuvio 9. WWW-palvelun toiminta palvelimella

Apachen jälkeen asennetaan MySQL-tietokantasovellus. Sitä tarvitaan Nagioksen vaatiman tietokannan ylläpitoa varten. MySQL asennetaan komennolla

```
apt-get install mysql-server php5-mysql
```

Tämä komento asentaa myös joukon lisäosia, joista MySQL:n toiminta (kuten myös Nagioksen) on riippuvainen. Asennuksen aikana pyydetään valitsemaan MySQL:n tietokannan root-tunnukselle salasana, mikä on suositeltavaa valita.

Viimeinen LAMP-kokoelman osa on PHP (Hypertext Preprocessor), joka asennetaan lisäosineen komennolla

```
apt-get install php5 libapache2-mod-php5 php5-mcrypt
```

3.2.3 **Välimuisti**

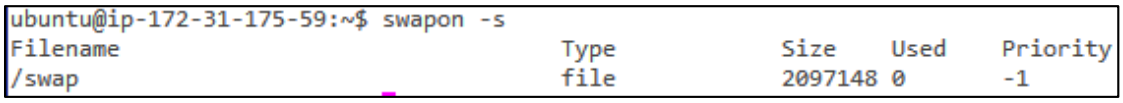
Näiden sovellusten lisäksi Nagios vaatii toimiakseen 2 gigatavun välimuistin, jonne Nagios tallentaa tiladataa. Mikäli sellaista ei ole aikaisemmin määritetty, se voidaan luoda komennoilla

```
dd if=/dev/zero of=/swap bs=1024 count=2097152 mkswap /swap &&
chown root. /swap && chmod 0600 /swap && swapon /swap
sh -c "echo /swap swap swap defaults 0 0 >> /etc/fstab"
sh -c "echo vm.swappiness = 0 >> /etc/sysctl.conf && sysctl -
p"
```

Välimuistin luomisen onnistuminen varmistetaan komennolla

```
swapon –s
```

jolloin tulosteen pitäisi olla kuvion 10 mukainen.



Kuvio 10. Välimuistin todentaminen

3.3 Nagioksen asennus

Nagios 4:n asennus alkaa monitorointisovelluksen rungon, Nagios Coren, asentamisella. Nagioksen toimintaa varten luodaan uusi käyttäjä ”nagios”, jolla Nagioksen prosessit suoritetaan. Samalla luodaan uusi käyttäjäryhmä ”nagcmd”, johon juuri luotu käyttäjä lisätään. Nämä tapahtuvat seuraavilla komennoilla:

```
useradd nagios
groupadd nagcmd
usermod -a -G nagcmd nagios
```

Käyttäjämäärityksen jälkeen asennetaan muutamia lisäkirjastoja, joiden avulla Nagios Core saadaan asennettua suoraan lähdekoodista. Nämä kirjastot asentuvat komennolla

```
apt-get install build-essential libgd2-xpm-dev openssl libssl-
dev xinetd apache2-utils unzip
```

Nagioksen lähdekoodi ladataan suoraan valmistajan verkkosivuilta. Uusimman version lataamisen varmistamiseksi kannattaa tarkastaa viimeisimmän version latauslinkki osoitteesta https://www.nagios.org/downloads/nagios-core/thanks/?t=1447082220. Tätä kirjoittaessa Nagioksen uusin versio on 4.1.1, joten siirrytään käyttäjän kotikansioon komennolla

```
cd ~
```

ja suoritetaan lähdekoodin lataus komennolla

```
curl -L -O
https://assets.nagios.com/downloads/nagioscore/releases/nagios
-4.1.1.tar.gz
```

Komento lataa Nagioksen lähdekoodipaketin tar-tiedostona. Tiedosto puretaan, minkä jälkeen siirrytään purettuun kansioon komennoilla

```
tar xvf nagios-4.1.1.tar.gz
cd nagios-4.1.1
```

Seuraavaksi suoritetaan Nagioksen esikonfigurointi komennolla, jossa määritetään käytettävä käyttäjä ja ryhmä

```
./configure --with-nagios-group=nagios --with-command-
group=nagcmd
```

, joka luo väliaikaisen konfiguraatiotiedoston kansioon sample-config/, ja tulosteena pitäisi olla kuvion 11 kaltainen ilmoitus, josta nähdään esimerkiksi käytettävä Apache www-palvelun konfiguraatiotiedosto (`/etc/http/conf.d`) sekä Nagioksen suorittamiseen käytettävä aiemmin luotu käyttäjä (Nagios user/group) ja ryhmä (command user/group).

```
*** Configuration summary for nagios 4.1.1 08-19-2015 ***:

General Options:
-------------------------
       Nagios executable:  nagios
       Nagios user/group:  nagios,nagios
      Command user/group:  nagios,nagcmd
            Event Broker:  yes
       Install ${prefix}:  /usr/local/nagios
   Install ${includedir}:  /usr/local/nagios/include/nagios
               Lock file:  ${prefix}/var/nagios.lock
  Check result directory:  ${prefix}/var/spool/checkresults
          Init directory:  /etc/init.d
 Apache conf.d directory:  /etc/httpd/conf.d
            Mail program:  /bin/mail
                 Host OS:  linux-gnu
         IOBroker Method:  epoll

Web Interface Options:
------------------------
                HTML URL:  http://localhost/nagios/
                 CGI URL:  http://localhost/nagios/cgi-bin/
Traceroute (used by WAP):


Review the options above for accuracy.  If they look okay,
type 'make all' to compile the main program and CGIs.
```

Kuvio 11. Nagioksen esikonfigurointi

Nagioksen lopullinen asennus tehdään useammalla komennolla. Ensimmäinen komento asentaa pääohjelman, CGI:n (Common Gateway Interface) ja HTML-tiedostot (Hyper Text Markup Language)

```
make install
```

Seuraava komento asentaa komentotiedoston sisältävälle kansiolle käyttöoikeudet

```
make install-commandmode
```

Seuraava komento ei ole pakollinen, mutta se asentaa Nagioksen kansioon valmiita pohjia valvottavien kohteiden konfigurointiin:

```
make install-config
```

Seuraavalla komennolla asennetaan Apache www-palveluun Nagioksen konfigurointitiedosto:

```
make install-webconf
```

Jos komento antaa kuvion 12 mukaista virhettä, Nagioksen asennus ei tunnista Apachen uudempaa versiota (Apache2) ja sen konfigurointitiedostojen sijaintia.

```
ubuntu@ip-172-31-175-59:~/nagios-4.1.1$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
/usr/bin/install: cannot create regular file '/etc/httpd/conf.d/nagios.conf': No such file or
directory
make: *** [install-webconf] Error 1
```

Kuvio 12. install-webconf -virhe

Sen sijaan voi joutua käyttämään komentoa

```
/usr/bin/install -c -m 644 sample-config/httpd.conf
/etc/apache2/sites-enabled/nagios.conf
```

jolla saadaan asennettua oikeaan sijaintiin Nagioksen www-palvelun konfigurointitiedosto.

Viimeinen Nagios Coren asennuskomento, jolla annetaan web-palvelulle oikeudet käskyttää Nagioksen palveluja web-selaimen kautta, on

```
usermod -G nagcmd www-data
```

Näiden toimenpiteiden jälkeen Nagioksen runko on asennettu. Palvelu ei ole vielä toimiva, vaan siitä puuttuvat täysin itse monitorointia tekevät lisäosat NRPE ja Nagios Plugins, jotka asennetaan luvuissa 3.4–3.5.

3.4 Nagios plugins-asennus

Nagios plugins (=lisäosat) sallivat varsinaiset laitteiden monitoroinnin. Ilman niitä, Nagios on käytännössä vain web-sivu, jota pystyy selailemaan. Lisäosien asennus alkaa siirtymällä käyttäjän kotikansioon, jonka jälkeen tarkastetaan uusin saatavilla oleva versio osoitteesta http://nagios-plugins.org/download/?C=M;O=D. Opinnäytetyötä tehdessä uusin saatavilla oleva versio on 2.1.1, joten paketti ladataan ja puretaan komennoilla

```
curl -L -O http://nagios-plugins.org/download/nagios-plugins-
2.1.1.tar.gz
tar xvf nagios-plugins-2.1.1.tar.gz
```

ja lopuksi siirrytään purettuun kansioon komennolla

```
cd nagios-plugins-2.1.1
```

Jälleen suoritetaan asennuksen esikonfigurointi komennolla

```
./configure --with-nagios-user=nagios --with-nagios-
group=nagios --with-openssl
```

, jossa määritetään käytettävä käyttäjä ja ryhmä sekä konfigurointi käyttämään OpenSSL-tukea. Nagios plugins-asennuksen koonti ja itse asennus käynnistetään komennoilla

```
make
make install
```

Näillä toimenpiteillä Nagios plugins on asennettu palvelimelle.

3.5 NRPE-asennus

NRPE:n avulla voidaan ajaa etäkoneilla Nagioksen komentoja. Jälleen tarkastetaan uusin saatavilla oleva versio osoitteesta http://sourceforge.net/projects/nagios/files/nrpe-2.x/ ja todetaan sen olevan 2.15. Ladataan ja puretaan se käyttäjän kotihakemistoon komennoilla

```
curl -L -O
http://downloads.sourceforge.net/project/nagios/nrpe-2.x/nrpe-
2.15/nrpe-2.15.tar.gz
tar xvf nrpe-2.15.tar.gz
```

Siirrytään purettuun kansioon ja suoritetaan esikonfigurointi komennolla

```
./configure --enable-command-args --with-nagios-user=nagios --
with-nagios-group=nagios --with-ssl=/usr/bin/openssl --with-
ssl-lib=/usr/lib/x86_64-linux-gnu
```

, jonka jälkeen pitäisi tulla kuvion 13 mukainen ilmoitus konfiguroinnin onnistumisesta. Siinä ilmoitetaan myös NRPE:n käyttämä portti, joka tulee sallia palvelimen ja verkon palomuurisäännöissä.

```
*** Configuration summary for nrpe 2.15 09-06-2013 ***:

 General Options:
 -------------------------
 NRPE port:    5666
 NRPE user:    nagios
 NRPE group:   nagios
 Nagios user:  nagios
 Nagios group: nagios


Review the options above for accuracy.  If they look okay,
type 'make all' to compile the NRPE daemon and client.
```

Kuvio 13. NRPE-esikonfiguroinnin tulos

Esikonfiguroinnin jälkeen suoritetaan asennuspakettien koonti ja lopullinen asennus komennoilla

```
make all
make install
make install-xinetd
make install-daemon-config
```

Komento ”make install-xinetd” määrittää käynnistysskriptin, jota muokataan haluamallaan editorilla (esimerkiksi nanolla) komennolla

```
nano /etc/xinetd.d/nrpe
```

ja lisätään rivin ”only_from = 127.0.0.1” perään Nagios-palvelimen sisäverkon IP-osoite, jolloin vain tästä osoitteesta tulevat pyynnöt voivat kommunikoida Nagios-palvelimen NRPE-lisäosan kanssa. Viimeisenä käynnistetään xinetd palvelu uudestaan komennolla

```
service xinetd restart
```

jonka jälkeen NRPE on asennettu.

3.6 Nagioksen esikonfigurointi

3.6.1 **Konfiguraatiotiedostojen järjestely**

Nagioksen konfiguraatiotiedostot on hyvä järjestellä haluamallaan tavalla. Tiedostojen sijaintia muutetaan muokkaamalla tiedostoa komennolla

```
nano /usr/local/nagios/etc/nagios.cfg
```

josta poistetaan risuaita riviltä

```
#cfg_dir=/usr/local/nagios/etc/servers
```

Tällä rivillä määritetään kansio, jossa monitoroitavien palvelimien konfiguraatiotiedostot sijaitsevat. Lopuksi kyseinen kansio luodaan komennolla

```
mkdir /usr/local/nagios/etc/servers
```

3.6.2 **Yhteystietojen konfigurointi**

Nagioksen yhteystiedot konfiguroidaan komennolla

```
nano /usr/local/nagios/etc/objects/contacts.cfg
```

Tiedostoon voidaan luoda uusia yhteystietoja tai yhteystietoryhmiä. Suositeltavaa on vaihtaa nagiosadmin-tunnuksen sähköpostiosoite riviltä

```
email          nagios@localhost        ; <<***** CHANGE THIS
TO YOUR EMAIL ADDRESS ******
```

haluamaansa osoitteeseen. Nagiosta suorittavalla palvelimella on oltava sähköpostin lähetysasetukset määritettynä, tämä prosessi on kuvattu tarkemmin luvussa 3.11.

3.6.3 **Apache WWW-palvelun konfigurointi**

Jotta Nagioksen WWW-sivu toimii halutulla tavalla, on Apachen moduuleista otettava käyttöön ”rewrite” ja ”cgi” moduulit. Tämä onnistuu komennoilla

```
a2enmod rewrite
a2enmod cgi
```

Lisäksi luodaan uusi käyttäjä htpasswd-työkalulla, jolla on oikeus päästä Nagioksen web-sivustoon, komennolla

```
htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

jonka yhteydessä asetetaan luodulle käyttäjälle salasana. Tämä salasana on tärkeää pitää tallessa, jotta päästään Nagioksen web-sivuille.

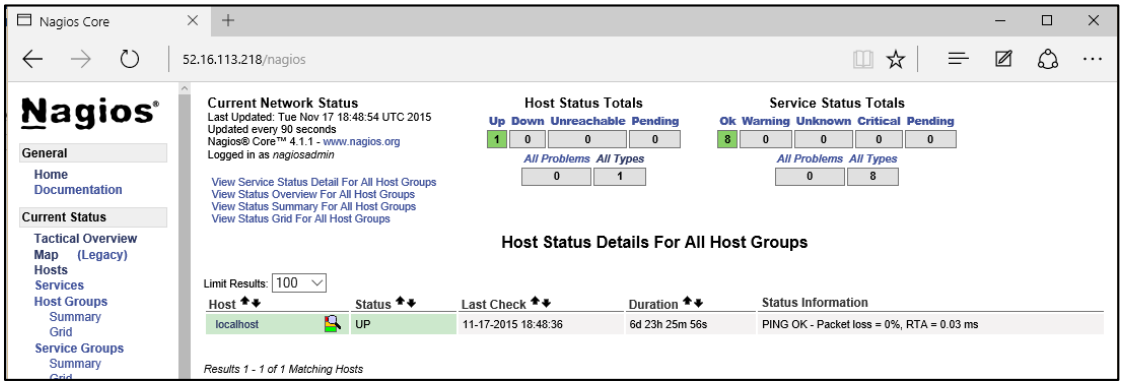
Luodaan symbolinen linkki Nagioksen konfiguraatiotiedostosta Apachen käytössä oleviin palveluihin komennolla

```
ln -s /etc/apache2/sites-available/nagios.conf
/etc/apache2/sites-enabled/
```

Lopuksi käynnistetään sekä Nagios että Apache-palvelut uudestaan ja asetetaan Nagios käynnistymään palvelimen käynnistyessä komennoilla

```
service nagios start
service apache2 restart
ln -s /etc/init.d/nagios /etc/rcS.d/S99nagios
```

Näiden konfiguraatioiden jälkeen Nagioksen peruspalvelu pitäisi olla saatavilla palvelimen IP-osoitteesta. Voimme todeta sen siirtymällä internetselaimella palvelimen silloiseen julkiseen IP-osoitteeseen. Kuviossa 14 on esitetty Nagioksen web-sivun ”Hosts”-näkymä. Siinä on asennusvaiheessa vain yksi seurattava palvelin, eli paikallinen localhost-palvelin, jolle Nagios on asennettu.



Kuvio 14. Nagioksen web-sivu

3.6.4 **Rajoitettu pääsy web-sivulle**

Nagioksen web-sivulle pääsyä voidaan rajoittaa myös yksinkertaisesti IP-osoitteen perusteella. Rajoitukset tehdään Nagioksen konfiguraatiotiedostoon `nagios.conf`, jota päästään muokkaamaan komennolla

```
nano /etc/apache2/sites-available/nagios.conf
```

Tiedostosta löytyvät seuraavat rivit

```
Order allow,deny
Allow from all
```

Nämä rivit sallivat yhteydet web-sivulle kaikista IP-osoitteista. Lisäämällä näiden rivien eteen risuaidan estetään rivien käyttö ja poistamalla risuaita rivien

```
#  Order deny,allow
#  Deny from all
#  Allow from 127.0.0.1
```

edestä, voidaan rajoittaa pääsyä web-sivulle. Nämä rivit esiintyvät tiedostossa kahteen kertaan, jolloin muutokset on tehtävä kumpaankin kohtaan. Ensimmäinen rivi kertoo järjestyksen, jolla rivejä luetaan. Ensin estetään ”deny” säännöllä, jonka jälkeen sallitaan erikoistapaukset ”allow” säännöllä. Ensin rivillä ”deny from all” estetään yhteydet kaikkialta, mutta sen jälkeen siirrytään lukemaan seuraavaa riviä, jossa määritetään ”allow from 127.0.0.1”. Tälle riville voidaan lisätä IP-osoitteita, joista yhteys web-sivulle on sallittu. Muutosten teon jälkeen sekä Nagios että Apache on käynnistettävä uudelleen.

3.7 Nagiosgraph-asennus

Nagiosgraph-lisäosalla saadaan Nagioksen keräämästä datasta graafisia raportteja. Se tallentaa Nagioksen keräämän tiedon RRD-tiedostoihin (Round-robin Database), jonka kautta saadaan tulostettua HTML-sivuja, joista saadaan tarkkaa lokitietoa monitoroitavan laitteen toiminnasta graafisessa muodossa.

Nagiosgraphin asennus aloitetaan luomalla uusi kansio ja lataamalla asennustiedostot kyseiseen kansioon komennoilla

```
mkdir -p /usr/local/src/nagiosgraph/
cd /usr/local/src/nagiosgraph/
wget
http://downloads.sourceforge.net/project/nagiosgraph/nagiosgra
ph/1.5.2/nagiosgraph-1.5.2.tar.gz
```

Seuraavaksi puretaan ladattu asennuspaketti ja siirrytään purettuun kansioon

```
tar -xf nagiosgraph-1.5.2.tar.gz
cd nagiosgraph-1.5.2
```

Asennus aloitetaan varmistamalla, että kaikki Nagiosgraphin vaatimat lisäosat ovat asennettuna komennolla

```
./install.pl --check-prereq
```

Opinnäytetyötä tehdessä kohdattiin kuvion 15 mukaiset virheet.

```
ubuntu@ip-172-31-175-59:/usr/local/src/nagiosgraph/nagiosgraph-1.5.2$ sudo ./install.pl --check-prereq
checking required PERL modules
  Carp...1.29
  CGI...3.63
  Data::Dumper...2.145
  Digest::MD5...2.52
  File::Basename...2.84
  File::Find...1.23
  MIME::Base64...3.13
  POSIX...1.32
  RRDs... ***FAIL***
  Time::HiRes...1.9725
checking optional PERL modules
  GD... ***FAIL***
  Nagios::Config... ***FAIL***
checking nagios installation
  found nagios exectuable at /usr/local/nagios/bin/nagios
  found nagios init script at /etc/init.d/nagios
checking web server installation
  found apache executable at /usr/sbin/apache2
  found apache init script at /etc/init.d/apache2

*** one or more problems were detected!

ubuntu@ip-172-31-175-59:/usr/local/src/nagiosgraph/nagiosgraph-1.5.2$
```

Kuvio 15. Nagiosgraph-esivaatimuksen virhe

Virheet johtuvat puuttuvista PERL:in (Practical Extraction and Report Language) kirjastosta, librrds-perl ja libgd-perl. Kirjastot eivät ole asentuneet esiasennuksen aikana. Ongelmat korjataan asentamalla kirjastot manuaalisesti komennoilla

```
apt-get install librrds-perl libgd-perl
```

Seuraava ongelma asennuksen tarkastuksessa oli kohta Nagios::Config, joka myös epäonnistui. Tässä tapauksessa PERL:stä oli jäänyt puuttumaan moduuli, jolla asennetaan Nagioksen konfigurointitiedot tietokantaan. Moduuli ladataan ja asennetaan PERL:n CPAN-tietokannasta (Comprehensive Perl Archive Network) komennolla

```
cpan Nagios::Config
```

Näiden korjaustoimenpiteiden jälkeen tarkastetaan Nagiosgraphin asennuksen esivaatimukset uudestaan, jolloin tulos pitäisi olla kuvion 16 mukainen.

```
ubuntu@ip-172-31-175-59:/usr/local/src/nagiosgraph/nagiosgraph-1.5.2$ sudo ./install.pl --check-prereq
checking required PERL modules
  Carp...1.29
  CGI...3.63
  Data::Dumper...2.145
  Digest::MD5...2.52
  File::Basename...2.84
  File::Find...1.23
  MIME::Base64...3.13
  POSIX...1.32
  RRDs...1.4007
  Time::HiRes...1.9725
checking optional PERL modules
  GD...2.50
  Nagios::Config...36
checking nagios installation
  found nagios exectuable at /usr/local/nagios/bin/nagios
  found nagios init script at /etc/init.d/nagios
checking web server installation
  found apache executable at /usr/sbin/apache2
  found apache init script at /etc/init.d/apache2
ubuntu@ip-172-31-175-59:/usr/local/src/nagiosgraph/nagiosgraph-1.5.2$
```

Kuvio 16. Nagiosgraph-esivaatimukset kunnossa

Nagiosgraphin asennus jatkuu aloittamalla asennus komennolla

```
./install.pl --layout standalone --prefix
/usr/local/nagiosgraph
```

Asennus kysyy asennussijainteja, joihin voidaan antaa oletusvastaus painamalla enter-painiketta. Asennuksen päätyttyä asennusohjelma kertoo vielä tehtäviä toimenpiteitä. Opinnäytetyötä tehdessä tuli tehdä seuraavat toimenpiteet:

1. Lisätä seuraavat rivit Nagioksen pääkonfiguraatiotiedostoon /usr/local/nagios/etc/nagios.cfg

```
# process nagios performance data using nagiosgraph
process_performance_data=1
service_perfdata_file=/tmp/perfdata.log
service_perfdata_file_template=$LASTSERVICECHECK$||$HOSTNAME$|
|$SERVICEDESC$||$SERVICEOUTPUT$||$SERVICEPERFDATA$
service_perfdata_file_mode=a
service_perfdata_file_processing_interval=30
service_perfdata_file_processing_command=process-service-
perfdata-for-nagiosgraph
```

2. Lisätä seuraavat rivit Nagioksen komentotiedostoon `/usr/local/nagios/etc/objects/commands.cfg`, joilla Nagiosgraph varsinaisesti ajetaan

```
# command to process nagios performance data for nagiosgraph
define command {
  command_name process-service-perfdata-for-nagiosgraph
  command_line /usr/local/nagiosgraph/bin/insert.pl
}
```

3. Määrittää Apache2 lukemaan myös Nagiosgraphin konfiguraatio lisäämällä seuraava rivi Apache2:n konfiguraatiotiedostoon `/etc/apache2/apache2.conf`

```
include /usr/local/nagiosgraph/etc/nagiosgraph-apache.conf
```

4. Tarkistaa Nagioksen konfiguraatiokomennolla, jonka tulee suoriutua läpi ilman ongelmia

```
/usr/local/nagios/bin/nagios -v
/usr/local/nagios/etc/nagios.cfg
```

Näiden lisäksi Ubuntun 14.04 versiossa tulee muokata Apache2:n käyttämää `/usr/local/nagiosgraph/etc/nagiosgraph-apache.conf` tiedostoa. Tiedostossa määritellään millä käyttäjillä on pääsy Nagiosgraphiin. Oletuksena tiedostolla on määritelty vapaa pääsy Nagiosgraphin sivustolle, joten se on muokattava tietoturvalliseksi. Tiedostoa muokataan komennolla

```
nano /usr/local/nagiosgraph/etc/nagiosgraph-apache.conf
```

ja vaihdetaan risuaidalla (#) merkatut rivit seuraaviksi

```
AuthName "nagiosadmin"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
Require valid-user
```

Näillä muutoksilla otetaan käyttöön käyttäjän autentikointi sivustolle siirtyessä. Sivusto on toteutettu Javascriptillä, joten se on sallittava erikseen. Samassa tiedostossa korvaa rivit

```
Order allow,deny
Allow from all
```

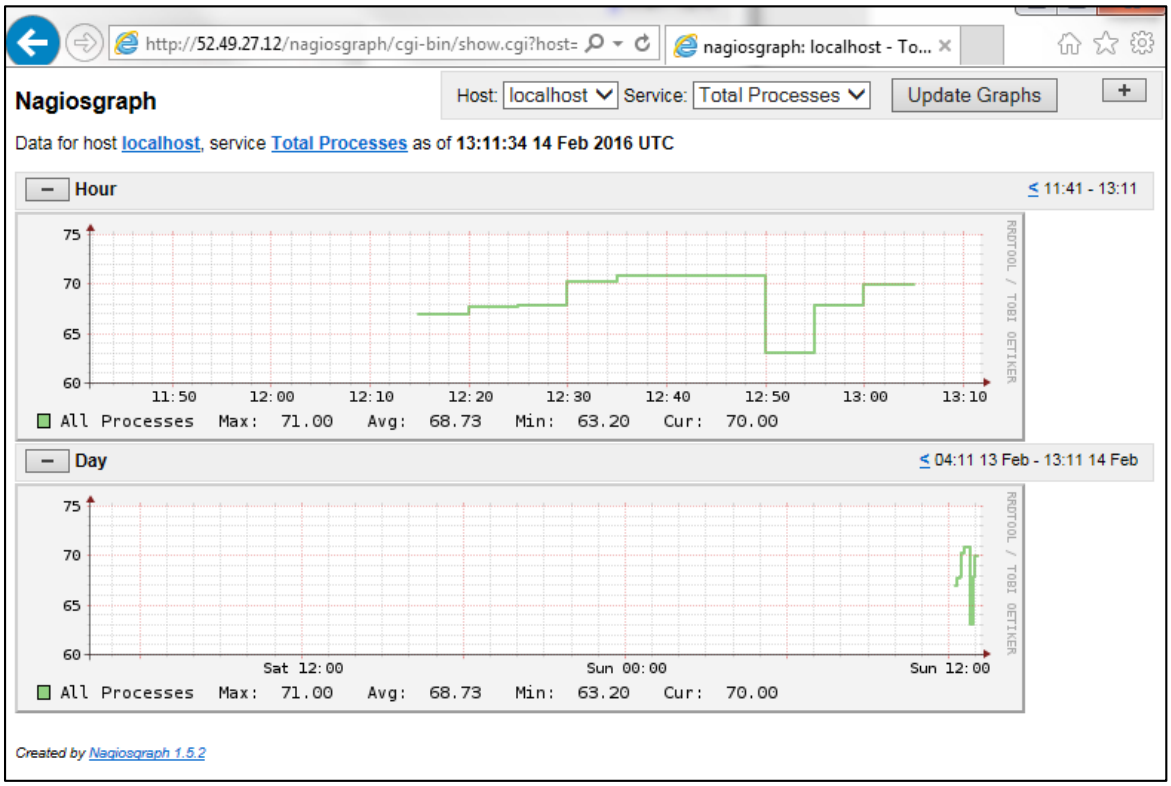
rivillä

```
Require all granted
```

Näillä muutoksilla on otettu käyttöön Nagiosgraphin käyttäjän autentikointi ja sallittu Javascriptin suorittaminen. Lopuksi käynnistetään Nagios ja Apache2 uudestaan komennoilla

```
service nagios restart
service apache2 restart
```

Nagiosgraphin dataan pääsee suoraan osoitteella http://palvelimen_ip-osoite/nagiosgraph/cgi-bin/show.cgi. Kuviossa 17 on esitetty näkymä Nagiosgraphin sivulta, jossa monitorointipalvelimelta on otettu esille kuvaajat, jotka kertovat kaikkien suoritettavien prosessien määrät viimeisen tunnin ja päivän ajalta.



Kuvio 17. Nagiosgraph-esimerkki

Nagiosgraphin kuvaajien linkit saadaan myös Nagioksen Services-välilehdelle. Myöhemmin kuvattavien palveluiden valvonnan yhteydessä määritellään palvelut, joita kullakin kohteella monitoroidaan. Lisäämällä myöhemmin määriteltävän palvelun alle muuttujan

```
action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
```

saadaan Services-välilehdelle painike, jota painamalla avautuu kyseisen palvelun tiedot Nagiosgraphilla.

Käyttömukavuuden kannalta Nagiosgraphin kuvaajat saadaan näkyville myös hiiren ylilennolla. Tämä onnistuu luomalla uusi tiedosto komennolla

```
nano /usr/local/nagios/share/ssi/common-header.ssi
```

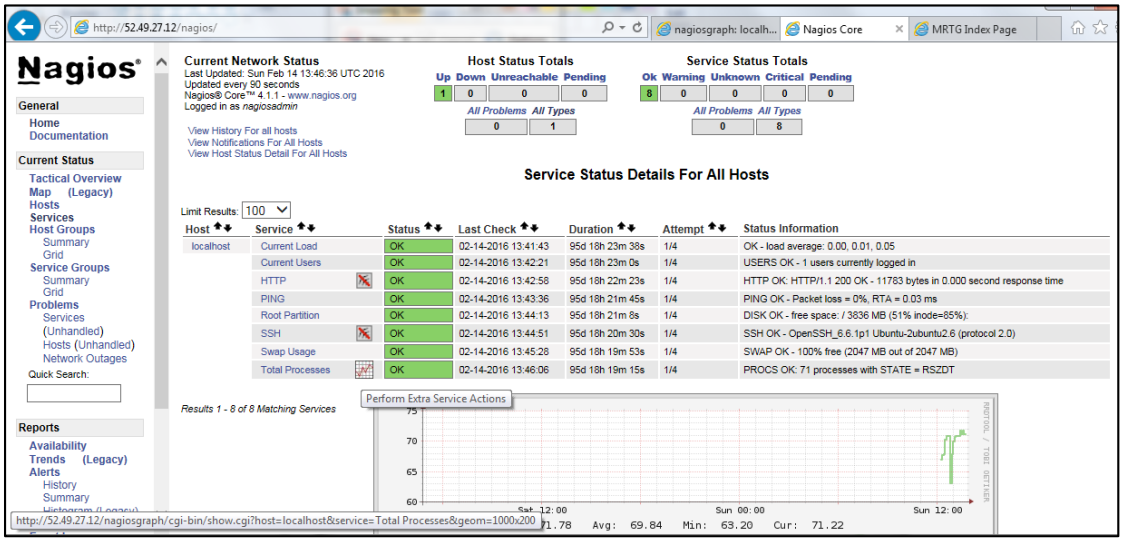
, johon lisätään rivi

```
<script type="text/javascript"
src="/nagiosgraph/nagiosgraph.js"></script>
```

ja käynnistetään jälleen Nagios uudestaan komennolla

```
service nagios restart
```

Tällöin näkymä Services-sivulla on kuvion 18 mukainen.



Kuvio 18. Nagiosgraph-kuvaaja Services-sivulla

3.8 Nagioksen statistiikan seuranta MRTG-työkalulla

Myös monitorointisovelluksen seuraaminen on tärkeää. Nagiokseen on sisäänrakennettu nagiostats-ominaisuus, jolla saadaan raakadataa toiminnasta. Tätä varten on kehitetty työkalu MRTG (Multi Router Traffic Grapher), joka muuttaa raakadatan seurattavampaan muotoon kuvaajiksi. Sen avulla pystytään varmistamaan Nagioksen tehokas toiminta, löytämään ongelma-alueet monitoroinnissa sekä seuraamaan konfiguraatiomuutosten tekemisten vaikutuksia monitorointiin. Sillä pystyy esimerkiksi seuraamaan, kuinka monta kohteen tarkastusta tehdään tunneittain sekä tarkastusten viiveitä.

MRTG:n konfiguraatiotiedosto Nagiosta varten on jo valmiiksi mukana Nagioksen asennuspaketissa. Ensimmäisenä MRTG-työkalu riippuvuuksineen on asennettava komennolla

```
apt-get install mrtg
```

Asennuksen aikana MRTG kysyy tiedoston käyttöoikeuksista. Tähän voidaan vastata kyllä.

Työkalu on otettava erikseen käyttöön, joka aloitetaan kopioimalla esimerkkitiedosto asennustiedostoista ja luomalla kansio kuvaajia ja tiedostoja varten komennoilla

```
cp /home/ubuntu/nagios-4.1.1/sample-config/mrtg.cfg
/usr/local/nagios/etc/
mkdir -p /usr/local/nagios/share/stats
```

Tämän jälkeen määritetään MRTG käyttämään kyseistä kansiota muokkaamalla konfiguraatiotiedostoa /usr/local/nagios/etc/mrtg.cfg ja lisäämällä tiedoston alkuun rivi

```
WorkDir: /usr/local/nagios/share/stats
```

Suoritetaan MRTG:n esiajo (antaa virheitä, ei vaikuta toimintaan) ja luodaan HTML-sivut komennoilla

```
env LANG=C /usr/bin/mrtg /usr/local/nagios/etc/mrtg.cfg
/usr/bin/indexmaker /usr/local/nagios/etc/mrtg.cfg --
output=/usr/local/nagios/share/stats/index.html
```

MRTG ei suoriudu automaattisesti, joten sitä varten on luotava ajoitettu työ, joka suoritetaan 5 minuutin välein. Luodaan uusi tiedosto komennolla

```
nano /etc/cron.d/mrtg-nagios
```

ja lisätään sinne rivi

```
*/5 * * * *  root  env LANG=C /usr/bin/mrtg
/usr/local/nagios/etc/mrtg.cfg
```

Näillä toimenpiteillä saadaan Nagioksen toiminnasta kuvaajia, jotka kertovat suorituskyvystä. Siirtymällä osoitteeseen http://palvelimen_ip-osoite/nagios/stats nähdään Nagioksen suorituskyvystä kuvaajia, joita on esitetty kuviossa 19.

Kuvio 19. Nagioksen suorituskyvyn kuvaajia (MRTG)

3.9 Nagiosgraph ja MRTG linkkien lisääminen Nagioksen etusivulle

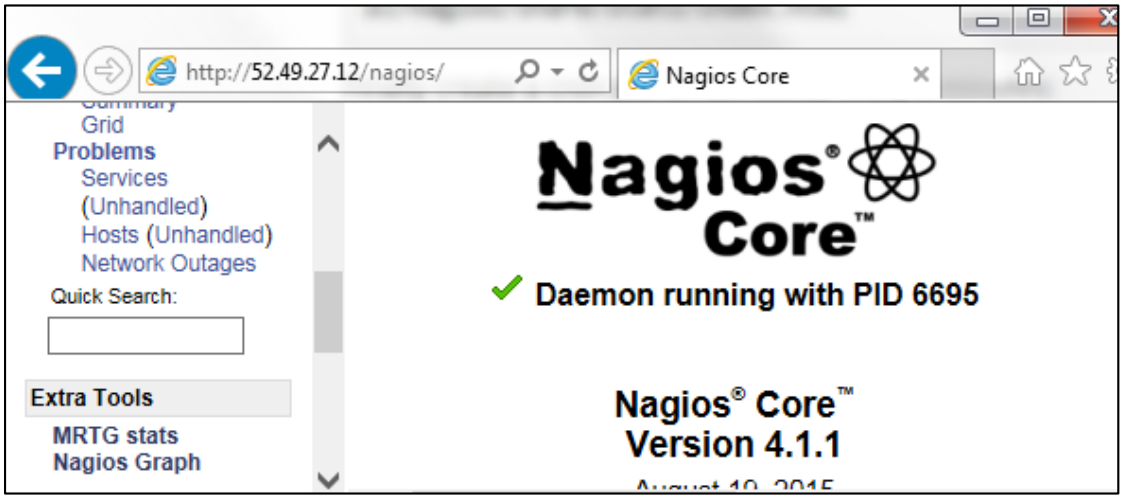
Käytön helpottamiseksi on järkevää lisätä linkit Nagiosgraphin ja MRTG:n sivuille siirtymiseen suoraan Nagioksen etusivulle. Muokataan Nagioksen sivuston PHP-tiedostoa komennolla

```
nano /usr/local/nagios/share/side.php
```

ja lisätään haluttuun kohtaan sivustolla

```
<div class="navsection">
    <div class="navsectiontitle">Extra Tools</div>
        <div class="navsectionlinks">
            <ul class="navsectionlinks">
                <li><a href="/nagios/stats" target="<?php echo
$link_target;?>">MRTG stats</a></li>
                <li><a href="/nagiosgraph/cgi-bin/show.cgi"
target="<?php echo $link_target;?>">Nagios Graph</a></li>
            </ul>
        </div>
    </div>
</div>
```

Tämä lisää kuvion 20 mukaisesti Nagioksen sivuvalikkoon Extra Tools-linkit MRTG:lle ja Nagiosgraphille.

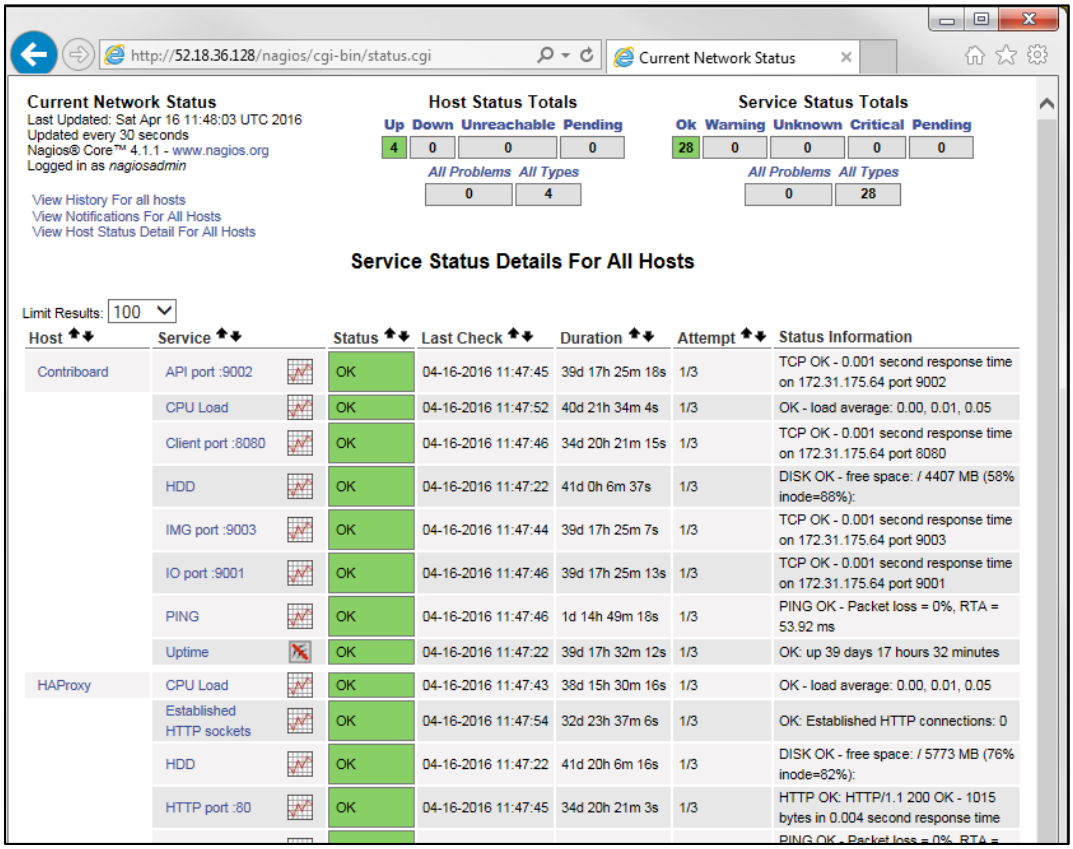


Kuvio 20. Pikalinkit sivuvalikossa

3.10 Yleisnäkymä palvelun tilasta

Opinnäytetön tavoitteena oli saada Nagioksen avulla helposti seurattava yleisnäkymä palvelun tilasta. Siitä tuli käydä ilmi päällä olevat ongelmat sekä kriittisimmät seurattavat palvelut. Ihmisen on vaikea seurata suurta määrää informaatiota, joten on tärkeää saada rajattua näkyviin vain kriittisimmät palvelut ja mahdolliset ongelmat, jotta pystytään reagoimaan muutenkin kuin sähköposti-ilmoituksen perusteella mahdollisiiin ongelmatilanteisiin.

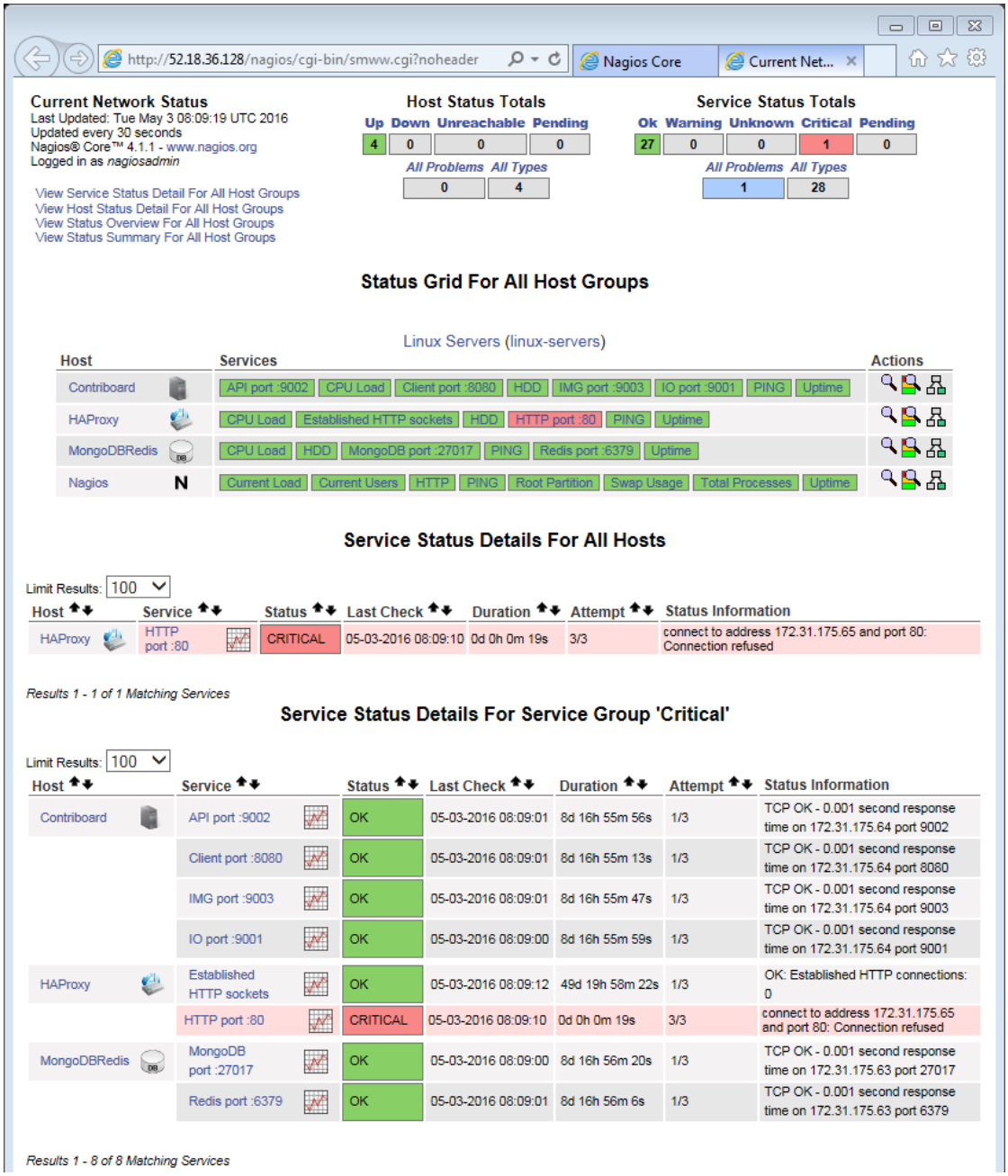
Nagioksessa on jo oletuksena graafinen näkymä, jota pystytään yleisnäkymän luomisessa hyödyntämään. Siirtymällä osoitteeseen http://IP-OSOITE/nagios/cgi-bin/status.cgi saadaan näkymä, joka listaa kaikki monitoroitavat palvelut, joita on esitetty kuviossa 21.



Kuvio 21. Nagios status.cgi graafinen näkymä

Hyödyntämällä status.cgi-näkymää voidaan rakentaa halutun näköinen näkymä palvelun yleistilasta. Nagios Exchangen verkkosivuilta (https://exchange.nagios.org) löytyy valmiiksi tehty skripti ”Show Me What’s Wrong” (SMWW). Skriptiin kuuluu smww.php ja smww.cgi. PHP-tiedosto kopioidaan tiedostoon `/usr/local/nagios/share/smww.php` ja CGI-tiedosto tiedostoon `/usr/local/nagios/sbin/smww.cgi`.

Jotta näkymästä saatiin halutunlainen, piti CGI-tiedostoa muokata. Lopullinen skripti on esitetty liitteessä 3. Yksinkertaisella skriptillä saadaan yhden ruudun näkymä, johon listataan monitoroitavien palveluiden yleistila (UP/DOWN), monitoroitavissa palveluissa esiintyvät CRITICAL/WARNING/UNKNOWN-tilat sekä kriittisten palveluiden tilat. Kuviossa 22 on esitetty näkymä, kun mon_testikone3-palvelimen HAProxy-ohjelma on pysäytetty manuaalisesti, eli vikatilannetta on simuloitu.



Kuvio 22. Ongelma HAProxy-palvelimella

Lisäksi yleisnäkymää varten on lisätty linkki Nagios-sivuston etusivulle sivupalkkiin. Lisäämällä tiedostoon `/usr/local/nagios/share/side.php` rivin

```
<li><a href="/nagios/cgi-bin/smww.cgi?noheader" target="<?php
echo $link_target;?>">Status View</a></li>
```

haluttuun kohtaan, saadaan yksinkertainen linkki näkymään luotua. Näkymä voidaan avata joko uuteen tai samaan välilehteen.

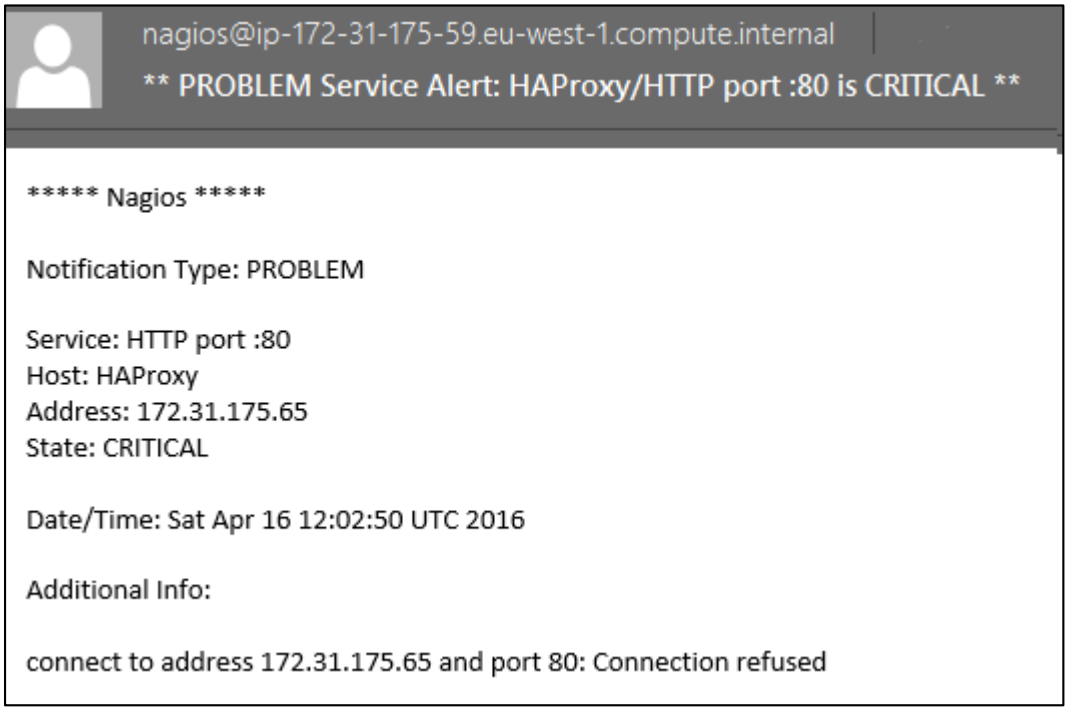
3.11 Sähköposti-ilmoitukset

3.11.1 **Yleistä**

Monitoroitavissa palveluissa olevista ongelmatilanteista on tärkeää saada tarvittaessa myös sähköpostilla ilmoitukset, mikäli luvussa 3.10 määritettyä näkymää ei pystytä jatkuvasti seuraamaan. Nagiokseen on perusasennuksessa jo valmiiksi konfiguroitu komennot, minkä mukaan sähköposteja lähetetään. Ne löytyvät tiedostosta `/usr/local/nagios/etc/objects/commands.cfg` ja ovat muotoa

```
# 'notify-host-by-email' command definition
define command{
        command_name    notify-host-by-email
        command_line    /usr/bin/printf "%b" "***** Nagios
*****\n\nNotification Type: $NOTIFICATIONTYPE$\nHost:
$HOSTNAME$\nState: $HOSTSTATE$\nAddress: $HOSTADDRESS$\nInfo:
$HOSTOUTPUT$\n\nDate/Time: $LONGDATETIME$\n" | /usr/bin/mail -
s "** $NOTIFICATIONTYPE$ Host Alert: $HOSTNAME$ is $HOSTSTATE$
**" $CONTACTEMAIL$
        }
# 'notify-service-by-email' command definition
define command{
        command_name    notify-service-by-email
        command_line    /usr/bin/printf "%b" "***** Nagios
*****\n\nNotification Type: $NOTIFICATIONTYPE$\n\nService:
$SERVICEDESC$\nHost: $HOSTALIAS$\nAddress:
$HOSTADDRESS$\nState: $SERVICESTATE$\n\nDate/Time:
$LONGDATETIME$\n\nAdditional Info:\n\n$SERVICEOUTPUT$\n" |
/usr/bin/mail -s "** $NOTIFICATIONTYPE$ Service Alert:
$HOSTALIAS$/$SERVICEDESC$ is $SERVICESTATE$ **" $CONTACTEMAIL$
        }
```

Printf-komento tulostaa sähköpostiviestin sisällön, joka sisältää muun muassa palvelun/palvelimen nimen, IP-osoitteen ja palvelun tilan. Mail-komento sisältää sähköpostiviestin otsikon, joka kertoo lyhyesti viestin sisällön. Jos viestin sisältöä halutaan muokata, on muutokset tehtävä näihin riveihin. Saapunut viesti esimerkkinä kuviossa 23.



Kuvio 23. Esimerkki hälytysviestistä Outlook-ohjelmassa

3.11.2 **Sähköpostiohjelman asentaminen ja konfigurointi**

Palvelimelle pitää olla asennettuna ohjelma, jolla pystytään lähettämään sähköpostiviestejä. Tähän käyttöön sopii esimerkiksi Postfix-ohjelma. Asennus aloitetaan komennolla

```
apt-get install mailutils
```

Komento asentaa palvelimelle sekä Postfixin, että muut sähköpostin lähettämiseen vaadittavat kirjastot. Asennuksen aikana joudutaan tekemään muutama valinta. Ensimmäiseen valintaruutuun valitaan Internet Site. Seuraavaksi asennusohjelma kysyy käytettävää domainia ”System mail name”-kohdassa. Jos käytettävissä on rekisteröity domain, voidsaan se valita tähän, muussa tapauksessa palvelimen hostname käy.

Lisäksi Postfixin konfiguraatiotiedostossa täytyy määrittää rajapinta, jonka kautta palvelin ottaa vastaan sähköpostiviestejä. Koska palvelimella ei ole tarvetta vastaanottaa viestejä, konfiguroidaan Postfix kuuntelemaan vain paikallista rajapintaa. Avataan konfiguraatiotiedosto muokkaamista varten komennolla

```
nano /etc/postfix/main.cf
```

Etsitään tiedostosta rivi, jossa lukee

```
inet_interfaces = all
```

ja muutetaan se muotoon

```
inet_interfaces = localhost
```

Tiedoston tallentamisen jälkeen Postfix-palvelu käynnistetään uudelleen komennolla

```
service postfix restart
```

Sähköpostin lähettämistä voi testata komennolla

```
printf "Viestin sisalto" | mail -s "Viestin otsikko"
osoite@domain.com
```

Jos testiviesti saapuu perille, toimivat myös Nagioksen lähettämät sähköpostiviestit.

3.11.3 **Nagioksen konfigurointi sähköpostihälytyksiä varten**

Nagioksen asennusvaiheessa kappaleessa 3.6.2 oli jo määritelty sähköpostiosoite nagiosadmin-käyttäjätunnukselle tiedostoon `/usr/local/nagios/etc/objects/contacts.cfg`. Samaan tiedostoon määritetään tarvittaessa myös muutkin yhteystiedot, joita palveluiden valvonnassa on määrä hyödyntää. Lisäksi tiedostoon voidaan määrittää yhteystietoryhmät, joilla yksittäisiä yhteystietoja saadaan niputettua.

Koska valvottavia palvelimia ja palveluita määrittäessä on käytetty Nagioksen valmiita pohjia, on niissä automaattisesti käytössä sähköposti-ilmoitukset. Yksittäiset palvelin- tai palvelukohtaiset määritykset voidaan tehdä palvelinkohtaiseen CFG-tiedostoon (esim. `/usr/local/nagios/etc/servers/mon_testikone3.cfg`). Hyvän käytännön mukaisesti kaikista valvottavista palveluista ei ole kannattavaa ottaa vastaan sähköpostitse ilmoituksia vuorokauden ympäri, vaan palvelun suorittamista varten kriittittisimmät osat on syytä ottaa tarkemman valvonnan alle.

4 Monitoroitavien palvelimien lisääminen Nagiokseen

4.1 Palvelimen konfigurointi monitorointia varten

Perusasennuksella Nagios monitoroi vain palvelinta, jolle se on asennettu, eli localhostia. Jokaiselle monitoroitavalle palvelimelle on asennettava aiemminkin mainitut Nagios plugins sekä NRPE server, joiden avulla monitorointipalvelin hakee tiedot asiakkailta. Nagios Plugins ja NRPE server asennetaan komennolla:

```
apt-get install nagios-plugins nagios-nrpe-server
```

Komento asentaa myös kaikki ohjelmien vaatimat riippuvuudet, joita voi olla paljonkin. Seuraavaksi määritetään NRPE:lle, mistä IP-osoitteesta se voi ottaa komentoja vastaan. Tässä on erittäin suositeltavaa käyttää sisäverkon IP-osoitetta, ettei internetin puolelta pystytä ajamaan haitallisia komentoja. IP-osoite määritetään komennolla

```
nano /etc/nagios/nrpe.cfg
```

lisäämällä riville allowed_hosts=127.0.0.1 Nagiosta suorittavan palvelimen sisäverkon IP-osoite, opinnäytetyössä rivi muutetaan muotoon:

```
allowed_hosts=127.0.0.1,172.31.175.59
```

Jos monitoroitavalla palvelimella on useampi verkkorajapinta, halutaan mahdollisesti rajoittaa NRPE:n toimintaa vain tiettyyn/tiettyihin rajapintoihin. Muokkaa tiedostossa riviä #server_address=127.0.0.1 niin, että siinä esiintyvä IP-osoite/osoitteet ovat rajapinnat, joista NRPE-komentoja halutaan ottaa vastaan, esimerkkinä palvelimen mon_testikone3:n sisäverkon IP-osoite, jonka jälkeen käynnistetään NRPE-prosessi uudelleen

```
server_address=172.31.175.65
service nagios-nrpe-server restart
```

4.2 Nagios-palvelimella tehtävä konfigurointi

Seuraavaksi monitoroitavan palvelimen tiedot on lisättävä monitorointipalvelimelle. Jokaista valvottavaa kohdetta kohden on tehtävä erillinen tiedosto. Aikaisempien

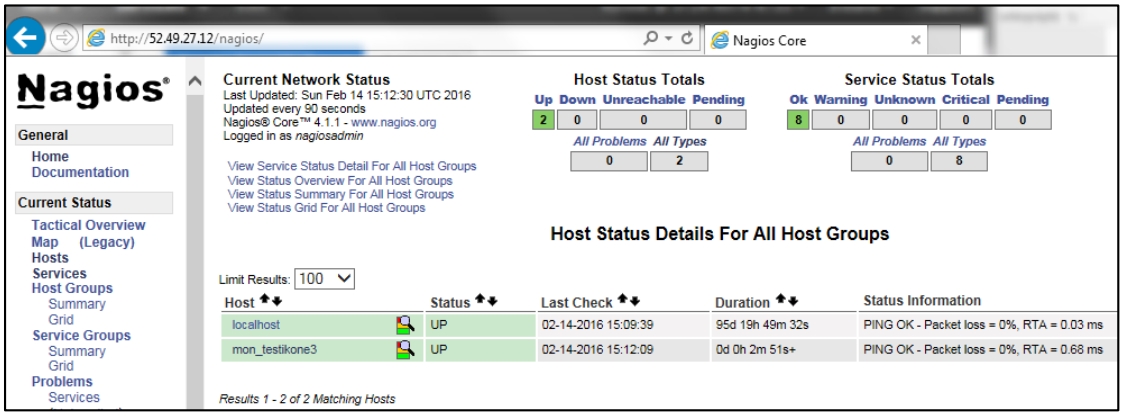
määrityksien mukaisesti tiedostot sijaitsevat kansiossa `/usr/local/nagios/etc/servers/`. Aloitetaan muokkaamalla/luomalla tiedosto komennolla

```
nano /usr/local/nagios/etc/servers/mon_testikone3.cfg
```

Lisätään tiedostoon perustiedot, joilla määritetään käytettävät pohja-asetukset (use), palvelimen nimi (host_name), WWW-sivulla näkyvä nimi (alias), IP-osoite (address), tarkastuksien yrityskerrat (max_check_attempts), valvonta-aika (check_period), huomautusväli (notification_interval) ja huomautusaika (notification_period)

```
define host {
        use                             linux-server
        host_name                       HAProxy
        alias                           HAProxy
        address                         172.31.175.59
        max_check_attempts              5
        check_period                    24x7
        notification_interval           30
        notification_period             24x7
}
```

Näillä perustoimenpiteillä voidaan valvoa vain kohteen olemassaoloa. Erillisten osa-alueiden valvonta on kerrottu myöhemmin. Käynnistetään Nagios-palvelu uudestaan ja todetaan monitoroinnin toiminta kuviosta 24.



Kuvio 24. mon_testikone3 lisätty monitoroitaviin kohteisiin

4.3 Palvelujen monitoroinnin määritys

Kappaleen 4.1 mukaiset toimenpiteet sallivat vain tietyn kohteen toimivuuden valvomisen. Yksittäisiä palveluja tai toimintoja ei vielä pystytä seuraamaan. Tässä vaiheessa opinnäytetyötä jokainen testiympäristössä käytettävä palvelin on lisätty monitoroitaviin kohteisiin, joten valvottavia palveluja voidaan alkaa määritellä.

Valvottavat palvelut määritellään Nagios-palvelimella konfiguraatiotiedostoihin palvelinkohtaisesti. Tiedostot sijaitsevat kansiossa `/usr/local/nagios/etc/servers` ja ne on syytä nimetä palvelimen nimellä, esimerkiksi `/usr/local/nagios/etc/servers/mon_testikone3.cfg`. Mon_testikone3 on hyvä esimerkkitapaus, koska sille asennettu HAProxy ohjaa HTTP-pyynnöt (Hypertext Transfer Protocol) Contriboardin eri osiin.

Palveluiden monitoroinnin määritys aloitetaan komennolla

```
nano /usr/local/nagios/etc/servers/mon_testikone3.cfg
```

Ensin määritellään yksinkertainen viiveen monitorointi, jota mitataan ping-pyynnöllä. Kasvanut viive kertoo yleensä ongelmista ensimmäisenä, joten se on erittäin tärkeä valvonnan kohde. Yksinkertaisimmillaan viiveen monitorointi onnistuu määrityksellä

```
define service{
        use                             generic-service
        host_name                       HAProxy
        service_description             PING
        servicegroups                   Network
        check_command                 check_nrpe!check_ping
}
```

Komennosta on jätetty pois opinnäytetyön siisteyden takia aiemmin mainittu action_url-komentorivi, jolla saadaan Nagiosgraphin kuvaaja Nagiokseen. Rivillä check_command määritellään suoritettava komento. Ilman check_nrpe-komentoa komento suoritettaisiin Nagios-palvelimelta käsin, joka antaa vääriä tuloksia palvelimien ollessa samassa sisäverkossa. Check_nrpe määrittää siis valvottavan komennon suorituksen host_name-rivillä määritetyssä kohteessa.

Valvottavan kohteen käytettävät komennot määritellään itse valvottavalla kohteella. Monitoroitaville palvelimille asennettu (kappale 4.1) NRPE on työkalu, jolla Nagios Plugins-komennot suoritetaan. Määritykset löytyvät monitoroitavalta palvelimelta tiedostosta `/etc/nagios/nrpe.cfg`. Tiedostoon on aiemmin konfiguroitu valvottavan kohteen ja monitorointipalvelimen IP-osoitteet. Tiedostoon lisätään rivi ping-komentoa varten, joka on esimerkiksi

```
command[check_ping]=/usr/lib/nagios/plugins/check_ping -H
178.16.180.2 -w 100,10% -c 250,50%
```

Eli monitorointipalvelimen konfiguraatiotiedostoon määritetty komento ”check_nrpe!check_ping” ajaa valvottavassa kohteessa nrpe.cfg tiedostosta komennon check_ping. Rivissä määritetään ensin käytettävä skriptitiedosto, jonka jälkeen annetaan tarkemmat määritykset komennolle

```
-H 178.16.180.2
-w 100,10%
-c 250,50%
```

-H vivulla annetaan ping-komennon kohde, -w vipu kertoo varoitusrajan (>100ms, >10% pakettihäviö) ja -c vipu kriittisen hälytyksen rajan (>250ms, >50% pakettihäviö).

Asetusten voimaantulemiseksi NRPE-prosessi on käynnistettävä uudelleen komennolla

```
service nagios-nrpe-server restart
```

Kaikki monitoroitavat palvelut on palvelinkohtaisesti määriteltävä konfiguraatiotiedostoihin. Tärkeintä on muistaa, että tietyt komennot on suoritettava check_nrpe-komennon kautta kohteella. Muutoin palvelun valvonta suoritetaan monitorointipalvelimella. Liitteessä 1 on esitetty kunkin monitoroitavan kohteen monitorointimääritykset (`/usr/local/nagios/etc/servers`-kansiosta) ja liitteessä 2 on esitetty kaikki NRPE-määritykset mon_testikone3-palvelimesta (`/etc/nagios/nrpe.cfg`-tiedostosta).

4.4 Lisätyt tarkastuskomennot

Opinnäytetyössä Nagioksen perusasennukseen on lisätty kaksi tarkastuskomentoa. Näistä check_uptime on käytössä kaikilla palveluun lisätyillä palvelimilla, jolla saadaan palvelimen ylläoloaika (uptime) valvonnan alle. Komento on jo valmiiksi tehty ja ladattu Nagios Exchangesta (https://exchange.nagios.org/directory/Plugins/System-Metrics/Uptime/check_uptime--2F-check_snmp_uptime/details). Komento ”asennetaan” kopioimalla lähdekoodi palvelimelle tiedostoon `/usr/lib/nagios/plugins/check_uptime.pl` ja sallimalla sen suorittaminen komennolla

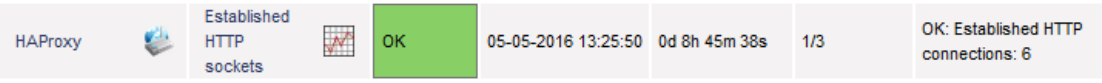
```
chmod a+x check_uptime.pl
```

Näiden toimenpiteiden jälkeen komento lisätään kohdepalvelimelle nrpe.cfg-tiedostoon sekä Nagios-palvelimelle kunkin monitoroitavan kohteen omaan cfg-tiedostoon.

Palvelimelle mon_testikone3 on lisäksi asennettu tarkastuskomento HTTP-yhteyksien määrien seuraamiseen. Tällä saadaan tarkka tieto, kuinka monta käyttäjää (tai avattua HTTP-yhteyttä) Contriboard-palvelussa on yhtäaikaisesti. Komento on määritetty mon_testikone3:n cfg-tiedostoon Nagios-palvelimella sekä kyseisellä palvelimella nrpe.cfg-tiedostoon. Molemmat tiedostot ovat liitteinä opinnäytetyössä. Komento nrpe.cfg:ssä on hyvin yksiselitteinen ja se menee seuraavasti

```
command[check_http_sockets]=/usr/lib/nagios/plugins/check_conn
ections3.pl -w 15 -c 30 -u root
```

eli komento käyttää tiedostoa `/usr/lib/nagios/plugins/check_connections3.pl`. Se on ladattu Nagios Exchangesta (https://exchange.nagios.org/directory/Plugins/Network-Connections,-Stats-and-Bandwidth/Check-Number-Of-Connections/details). Komennossa määritellään käytettävä tiedosto, sekä sen lisäksi varoitus (-w 15) ja kriittisen (-c 30) rajat, joiden ylityksestä lähetetään sähköpostilla hälytysviesti. Komennon lopussa määritetään miltä käyttäjältä HTTP-yhteyksiä seurataan (-u root). Skriptillä saadaan kuvion 25 mukainen näkymä Nagioksen valvontaan.



Kuvio 25. Avoimet HTTP-yhteydet

5 Nagioksen automatisointi

5.1 Docker-kontti

5.1.1 **Docker-kontin käyttöönotto**

Nagioksen asennusta voidaan automatisoida esimerkiksi käyttämällä Docker-nimistä työkalua. Sen avulla voidaan palvelimen sisällä suorittaa erinäisissä konteissa sovelluksia, ilman niiden varsinaista asennusta palvelimelle. Tässä opinnäytetyössä on käytetty valmista konttia, joka löytyy osoitteesta https://hub.docker.com/r/quantumobject/docker-nagios/.

Docker asennetaan yhdellä komennolla, joka suorittaa asennuksen bash-skriptinä osoitteesta https://get.docker.com/

```
wget -qO- https://get.docker.com/ | sh
```

Dockerin asentamisen jälkeen suoritetaan valmis kontti komennolla

```
docker run -d -p 80:80 quantumobject/docker-nagios
```

Komento lataa Dockerin tietokannasta oikean kontin, käynnistää sen ja ohjaa isäntäkoneelle saapuvat HTTP-yhteydet kontin sisälle porttiin 80. Koska opinnäytetyössä isäntäkoneelle on jo asetettu Apache2-asennus kuuntelemaan porttia 80, joudutaan palvelimella vaihtamaan kuunneltava portti esim. 8080-porttiin, jotta liikenne ohjautuu kontin sisällä olevaan Nagios-palveluun. Muutos tehdään Apache2:n konfiguraatiotiedostoon `/etc/apace2/ports.conf` vaihtamalla rivi

```
Listen 80
```

muotoon

```
Listen 8080
```

ja käynnistämällä Apache2 uudelleen komennolla

```
service apache2 restart
```

Kontin CLI:hin (Command Line Interface) pääsee isäntäkoneelta käyttämällä komentoa

```
docker exec -it container_id /bin/bash
```

container_id on tunniste, joka generoidaan konttia luodessa. Sen voi tarkastaa isäntäkoneelta komennolla

```
docker ps -a
```

Ennen kuin kontin sisällä pystytään käyttämään tekstinkäsittelyohjelmia, kuten nano, on kontissa suoritettava komento

```
export TERM=xterm
```

Oletuksena palveluun pääsee kirjautumaan käyttäjätunnuksella nagiosadmin ja salasanalla admin, joten on ehdottoman tärkeää vaihtaa tunnuksen salasana komennolla

```
htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

Konttiin asennettuihin ohjelmiin kuuluu Apache2, Nagios sekä NRPE, joten kontin sisällä voidaan aloittaa palvelimen konfigurointi opinnäytetyön kappaleesta 3.7. Kontti sisältää jo valmiiksi Postfix-sähköpostiohjelman asennuksen, mutta joidenkin osien puuttuessa se ei ole käyttövalmis. Suorittamalla Postfixin asennuksen uudestaan opinnäytetyön luvun 3.11.2 mukaisesti, saadaan sähköpostiviestit jälleen toimimaan.

5.1.2 **Nagioksen siirto palvelimelta Docker-konttiin**

Opinnäytetyössä haluttiin testata, onko mahdollista siirtää palvelimelle asennettu Nagios suoritettavaksi Docker-kontin sisältä. Testi aloitettiin luomalla kontti luvun 5.1.1 mukaisesti. Tämä ottaa käyttöön Nagioksen perusasennuksen, sisältäen NRPE ja Nagios Plugins-asennukset.

Kaikki Nagios 4.1.1:n asennus- ja konfigurointitiedostot sijaitsevat kansiossa `/usr/local/nagios/`. Kansio voidaan kopioida isäntäkoneelta konttiin väliaikaiseen sijaintiin komennolla

```
docker cp /usr/local/nagios container_id:/usr/local/nagioscopy
```

Tämän jälkeen kontissa olevat konfiguraatiotiedostot on korvattava isäntäkoneella kopioiduilla. Siirrytään konttiin ja sallitaan terminaalin käyttö komennoilla

```
docker exec -it container_id /bin/bash
export TERM=xterm
```

Ensin kontilla suoritettava Nagios-prosessi on pysäytettävä. Prosessia ei kuitenkaan tunnisteta, ennen kuin se on kertaalleen manuaalisesti käynnistetty. Komennoilla

```
service nagios start
service nagios stop
```

prosessi saadaan hallitusti ajettua alas. Tämän jälkeen voidaan korvata kontissa oletuksena olleet asetukset isäntäkoneelta tuoduilla. Siirrytään kansioon ja poistetaan alkuperäiset konfiguraatiot komennoilla

```
cd /usr/local/
rm -rf nagios
```

jonka jälkeen yksinkertaisesti vaihdetaan kopioidun kansion nimi komennolla

```
mv nagioscopy/ nagios
```

Docker cp-komento ei kopioi kansioiden tai tiedostojen omistajaa, vaan jättää ne root-käyttäjälle, joten ne on vielä muutettava. Nagios suorittaa prosessejaan nagios-käyttäjätunnuksella, joten komennolla

```
chown -R nagios:nagios nagios
```

voidaan muuttaa sijainnin `/usr/local/nagios/`-kansion, sen alikansioiden ja tiedostojen omistajaksi käyttäjä ja ryhmä nagios. Tämän jälkeen Nagioksen prosessi käynnistetään uudestaan komennolla

```
service nagios start
```

jolloin isäntäkoneella olleet konfiguraatiot tulevat käyttöön kontin sisällä suoritettavaan Nagiokseen. Palvelua testatessa huomattiin kuitenkin, että Nagios toisinaan antaa käyttöoikeusvirhettä sivustolla. Suositeltavampaa on siis alusta pitäen tehdä konfiguraatiomuutokset manuaalisesti opinnäytetyön mukaisesti, jotta vastaavanlaisilta käyttöoikeusongelmilta vältyttäisiin.

5.2 Monitoroitavien kohteiden asennuksen automatisointi

5.2.1 **Monitoroitavan kohteen skripti**

Asennuksen yksinkertaistamiseksi voidaan monitoroitavien kohteiden asennusta automatisoida. Kirjoittamalla yksinkertainen bash-skripti voidaan asennukset tehdä periaatteessa yhdellä komennolla. Opinnäytetyötä varten kirjoitettu skripti on esitetty liitteessä 4. Skripti on tehty Debian-pohjaisia (esim. Ubuntu) käyttöjärjestelmiä varten.

Skripti tallennetaan kohdepalvelimelle esimerkiksi nimellä asennus.sh. Tämän jälkeen tiedosto asetetaan suoritettavaksi komennolla

```
chmod a+x asennus.sh
```

, jonka jälkeen se voidaan suorittaa komennolla

```
./asennus.sh
```

Skripti ensin päivittää tietokannan ja asentaa apt-get -komennolla Nagios Plugins ja NRPE Server-ohjelmat. Tämän jälkeen käyttäjältä kysytään tietoja: Nagios-palvelimen IP-osoite ja check_ping-komentoa varten kohde-IP. Seuraavaksi skripti hakee tiedon levyjärjestelmän nimestä check_hda1-komentoa varten ja lisää komennon nrpe.cfg-tiedostoon, sekä hakee palvelimen IPv4-osoitteen hostname -i-komennolla ja lisää sen nrpe.cfg tiedostoon oikeassa muodossa.

Lopuksi skriptissä poistetaan tarpeettomia tietoja konfiguraatiotiedostosta ja lisätään aiemmin mainittu check_ping-komento konfiguraatioon. Viimeisenä NRPE-palvelu

käynnistetään uudestaan, jotta muutokset tulevat voimaan. Tämän jälkeen voidaan siirtyä opinnäytetyön lukuun 5.2.2, jossa käydään läpi Nagios-palvelimen puolelle kirjoitettu skripti.

5.2.2 **Nagios-palvelimen skripti**

Myös Nagios-palvelimella tehtävään automatisoituun konfigurointiin on opinnäytetyössä tehty skripti. Tämä skripti on esitetty liitteessä 5.

Kuten myös luvun 5.2.1 skriptissä käyttäjältä kysytään ensin palvelimen lisäämiseen tarvittavia tietoja, kuten palvelimen ”alias”, eli Nagioksessa esiintyvä nimi sekä sen IPv4-osoite. Näiden tietojen antamisen jälkeen skripti luo kansioon /usr/local/nagios/etc/servers käyttäjän antamalla alias-nimellä cfg-tiedoston, johon lisätään käytettävä pohja (linux-server) sekä kaksi palvelua valvottavaksi (check_disk ja check_ping). Toimenpiteiden jälkeen skripti käynnistää Nagios-prosessin uudestaan, jonka jälkeen palvelin on lisätty Nagioksen valvonnan piiriin.

6 Tulokset ja pohdinta

6.1 Tulokset

Opinnäytetyön tärkeimpänä tuloksena voidaan pitää yksityiskohtaisten ohjeiden laatimista ilmaisen Nagios Open source-monitorointisovelluksen asentamiseen Linux-ympäristöön. Ohjeissa on otettu huomioon mahdolliset ilmi tulevat virheet ja laadittu erilliset ohjeet näitä tilanteita varten. Lisäksi työssä on yhdessä toimeksiantajan kanssa sovittu, minkälainen muokattu yleisnäkymä palvelun tilasta oli tavoitteena saada, jotta tilanteen seuraaminen ihmiselle olisi helpompaa.

Työssä on myös tutkittu Contriboard-palvelussakin hyödynnettyä Docker-ohjelmistoa ja sen käyttämistä, jolla monitorointisovellus voidaan suorittaa tarvittaessa erillisen kontin sisällä. Lopputuloksena kuitenkaan kontittamisesta ei saatu hyödynnettyä kaikkea tarvittavaa, joten tässä puolessa opinnäytetyössä olisi ollut kehitettävää.

Lopulta opinnäytetyössä on käyty läpi myös Nagios-monitoroinnissa hyödynnettävää automatisointia kirjoittamalla kaksi eri skriptiä monitoroitavan kohteen lisäämiseksi. Näiden skriptien avulla voidaan hieman helpottaa ja nopeuttaa uusien laitteiden lisäämistä valvonnan piiriin. Skriptit on kuitenkin tehty toimimaan vain Debian-pohjaisilla Linux-käyttöjärjestelmillä.

6.2 Pohdinta

Opinnäytetyön valmistuminen viivästyi kuudella kuukaudella, mikä osittain vaikutti myös motivaatioon. Henkilökohtaisen elämän kiireet aiheuttivat aikatauluongelmia, eikä työtä saatu valmistumaan ajoissa. Viivästymisestä kuitenkin sovittiin yhdessä sekä opinnäytetyön ohjaajan, että toimeksiantajan kanssa.

Opinnäytetyössä eniten hyötyä oppimisen kannalta oli tiivis tutustuminen Linux-käyttöjärjestelmän konfigurointiin terminaalikomentojen kautta. Työssä jouduttiin selvittämään useita eri ongelmakohtia asennusvaiheessa sekä tutustumaan eri ohjelmointikieliiin, jotta palvelusta saatiin muokattua toimeksiantajan hyväksymän kaltainen.

Lisäksi työssä sai hyvän käsityksen verkon monitoroinnista ja sen tärkeydestä palvelun tuottamisessa, josta voi olla hyötyä tulevalla työuralla. Myös tutustuminen sovellusten kontittamiseen auttaa virtuaalikoneiden ja konttien vertailussa, tulevaisuudessa tästäkin taidosta ja tiedosta voi olla hyötyä.

Sen lisäksi, että opinnäytetyö on tehty Contriboard-ympäristöön, voi tehtyjä asennusohjeita käyttää pohjana myös muuhun ympäristöön Nagiosta asentaessa, jolloin kirjoitetusta ohjeista voi olla apua myös kolmansille osapuolille.

Lähteet

About Nagios. N.d. Tietoa Nagioksesta. Viitattu 17.4.2016. https://www.nagios.org/about/

AWS IP Address Ranges. 2015. Amazon Web Services julkisten IP-osoitteiden alueet. Viitattu 4.11.2015. http://docs.aws.amazon.com/general/latest/gr/aws-ip-ranges.html

Contriboard. N.d. Digitize ideas together while brainstorming. Viitattu 9.9.2015. http://n4sjamk.github.io/contriboard/

N4S-ohjelma. N.d. N4S-ohjelma: suomalaiset ohjelmistoyritykset nopeuttavat digitaalista taloutta. Viitattu 9.9.2015. http://www.n4s.fi/fi/

N4S@JAMK. N.d. N4S@JAMK etusivu. Viitattu 9.9.2015. https://n4sjamk.github.io/

Nagios Core Documentation. N.d. Nagios Core-ohjelmiston dokumentaatio. Viitattu 17.4.2016. https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/toc.html

Network Monitoring Best Practices. N.d. Verkon monitoroinnin parhaat käytännöt. Viitattu 16.5.2016. http://www.solarwinds.com/network-monitoring-best-practices.aspx

Network Monitoring Definition and Solutions. 2007. CIO:n artikkeli verkon monitoroinnin määritelmästä ja ratkaisuista. Viitattu 16.5.2016. http://www.cio.com/article/2438133/networking/network-monitoring-definition-and-solutions.html

Understand the architecture. N.d. Docker Enginen toimintatapa. Viitattu 18.5.2016. https://docs.docker.com/engine/understanding-docker/

What is Docker? N.d. Dockerin lyhyt esittely. Viitattu 18.5.2016. https://www.docker.com/what-docker

Liitteet

Liite 1. Valvottavien kohteiden määritykset

mon_testikone3.cfg

```
define host {
        use                             linux-server
        host_name                       HAProxy
        alias                           HAProxy
        address                         172.31.175.65
        max_check_attempts              5
        check_period                    24x7
        notification_interval           30
        notification_period             24x7
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       HAProxy
        service_description             PING
        servicegroups                   Network
        check_command                   check_nrpe!check_ping
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       HAProxy
        service_description             HTTP port :80
        check_command                   check_http
        check_interval                  30
        servicegroups                   Network,Critical
#        notifications_enabled           0
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       HAProxy
        service_description             HDD
        check_command                   check_nrpe!check_hda1
        check_interval                  60
        servicegroups                   Hardware
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
```

```
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       HAProxy
        service_description             CPU Load
        check_command                   check_nrpe!check_load
        check_interval                  30
        servicegroups                   Hardware
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
        host_name                       HAProxy
        service_description             Established HTTP
sockets
        check_command
check_nrpe!check_http_sockets
        servicegroups                   Network,Critical
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}
define service{
        use                             generic-service
; Name of service template to use
        host_name                       HAProxy
        service_description             Uptime
        check_command
check_nrpe!check_uptime
        servicegroups                   Uptime
        check_interval                  600
        notifications_enabled           0
}
```

mon_testikone2.cfg

```
define host {
        use                             linux-server
        host_name                       MongoDBRedis
        alias                           MongoDBRedis
        address                         172.31.175.63
        max_check_attempts              5
        check_period                    24x7
        notification_interval           30
        notification_period             24x7
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       MongoDBRedis
        service_description             PING
        servicegroups                   Network
        check_command                   check_nrpe!check_ping
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       MongoDBRedis
        service_description             Redis port :6379
        check_command                   check_tcp!6379
        check_interval                  30
        servicegroups                   Network,Critical
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       MongoDBRedis
        service_description             MongoDB port :27017
        check_command                   check_tcp!27017
        check_interval                  30
        servicegroups                   Network,Critical
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
```

```
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       MongoDBRedis
        service_description             HDD
        servicegroups                   Hardware
        check_interval                  60
        check_command                   check_nrpe!check_hda1
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       MongoDBRedis
        service_description             CPU Load
        check_interval                  30
        check_command                   check_nrpe!check_load
        servicegroups                   Hardware
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}
define service{
        use                             generic-service
; Name of service template to use
        host_name                       MongoDBRedis
        service_description             Uptime
        check_command
check_nrpe!check_uptime
        check_interval                  600
        servicegroups                   Uptime
        notifications_enabled           0
       }
```

mon_testikone1.cfg

```
define host {
        use                             linux-server
        host_name                       Contriboard
        alias                           Contriboard
        address                         172.31.175.64
        max_check_attempts              5
        check_period                    24x7
        notification_interval           30
        notification_period             24x7
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             PING
        servicegroups                   Network
        check_command                   check_nrpe!check_ping
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             Client port :8080
        check_command                   check_tcp!8080
        check_interval                  30
        servicegroups                   Network,Critical
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             IO port :9001
        check_command                   check_tcp!9001
        check_interval                  30
        servicegroups                   Network,Critical
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
```

```
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             API port :9002
        check_command                   check_tcp!9002
        check_interval                  30
        servicegroups                   Network,Critical
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             IMG port :9003
        check_command                   check_tcp!9003
        check_interval                  30
        servicegroups                   Network,Critical
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouseOver='showGraphPopup(this)'
```

```
onMouseOut='hideGraphPopup()' rel='/nagiosgraph/cgi-
bin/showgraph.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
}
define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             HDD
        check_interval                  60
        check_command                   check_nrpe!check_hda1
        servicegroups                   Hardware
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouse$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             CPU Load
        check_interval                  30
        check_command                   check_nrpe!check_load
        servicegroups                   Hardware
        action_url      /nagiosgraph/cgi-
bin/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$&geom=1000x2
00' onMouse$
}

define service{
        use                             generic-service
; Name of service template to use
        host_name                       Contriboard
        service_description             Uptime
        check_command
check_nrpe!check_uptime
        servicegroups                   Uptime
        check_interval                  60
        notifications_enabled           0
}
```

Liite 2. mon_testikone3 NRPE-komennot

```
command[check_users]=/usr/lib/nagios/plugins/check_users -w 5
-c 10
command[check_load]=/usr/lib/nagios/plugins/check_load -w
0.7,0.6,0.5 -c 0.9,0.8,0.7
command[check_hda1]=/usr/lib/nagios/plugins/check_disk -w 20%
-c 10% -p /dev/xvda1
command[check_zombie_procs]=/usr/lib/nagios/plugins/check_proc
s -w 5 -c 10 -s Z
command[check_total_procs]=/usr/lib/nagios/plugins/check_procs
-w 150 -c 200
command[check_ping]=/usr/lib/nagios/plugins/check_ping -H
178.16.180.2 -w 100,10% -c 250,50%
command[check_http_sockets]=/usr/lib/nagios/plugins/check_conn
ections3.pl -w 15 -c 30 -u root
command[check_uptime]=/usr/lib/nagios/plugins/check_uptime.pl
```

Liite 3. smww.cgi-tiedosto

```
#!/bin/sh

###
### SMWW - Show Me Whats Wrong Ver 1.2
###      by John Tysko, tysko@ohio.edu
###        update 2014/06/05 servicestatustypes changed from
16 to 28
###
###  uses standad nagios interface to display 3 sections -
###      network problems
###      host problems
###      services problems (not on a host that is down)
###

BINDIR=`dirname $0`

REQUEST_METHOD="GET"
export REQUEST_METHOD

#Vaihtaa sivun titlen Network Outages -> Service Overview
sed -e '1,$s/Network Outages/Service Overview/'

#Näyttää kaikki monitoroitavat palvelut kaikilta palvelimilta
grid näkymässä
QUERY_STRING='hostgroup=all&style=grid'
export QUERY_STRING
$BINDIR/status.cgi | grep -v '^Cache-Control:' | grep -v
'^Pragma: no-cache' |  \
 grep -v '^Last-Modified' | grep -v '^Expires:' | grep -v
'^Content-type:'

#Näyttää palvelut, jotka tilassa 'CRITICAL/UNKNOWN/WARNING' ja
palvelimen tila 'UP/PENDING'
QUERY_STRING='host=all&type=detail&servicestatustypes=28&hosts
tatustypes=3&serviceprops=42&noheader'
export QUERY_STRING
$BINDIR/status.cgi | grep -v '^Cache-Control:' | grep -v
'^Pragma: no-cache' | grep -v '^Refresh: ' |  \
 grep -v '^Last-Modified' | grep -v '^Expires:' | grep -v
'^Content-type:'

#Näyttää palvelut servicegroupissa 'Critical' kaikilta
palvelimilta
QUERY_STRING='host=all&servicegroup=Critical&style=details&noh
eader'
export QUERY_STRING
$BINDIR/status.cgi | grep -v '^Cache-Control:' | grep -v
'^Pragma: no-cache' | grep -v '^Refresh: ' |  \
 grep -v '^Last-Modified' | grep -v '^Expires:' | grep -v
'^Content-type:'
```

Liite 4. Monitoroitavan kohteen asennusskripti

```
#! /bin/bash
#Skripti, joka asentaa linux-palvelimelle Nagios Plugins ja NRPE ja
konfiguroi NRPE:n annettujen määreiden mukaisesti

#Step 1: apt-get update && install

apt-get update && apt-get install nagios-plugins nagios-nrpe-server
echo y


###Step 2: Kerätään tiedot

#Kysytään Nagios-palvelimen IP-osoite
printf "\nNagios-palvelimen IP? (IPv4, esim. 192.168.0.1)\n"
read nagiosip

#Kysytään check_ping-komennon kohde IP-osoite
printf "\nValitse PING-kohde (IPv4, esim. 192.168.0.1):\n"
read kohde

#Selvitetään df -h / | grep /-komennolla levyjärjestelmän nimi
while read -r levyasema junk; do
#Tulosta nrpe.cfg-tiedostoon check_hda1-komento
       printf '%s\n' "com-
mand[check_hda1]=/usr/lib/nagios/plugins/check_disk -w 20% -c 10% -p
$levyasema" >> /etc/nagios/nrpe.cfg
done < <(df -h / | grep /)

#Selvitetään hostname -i-komennolla palvelimen oma IP-osoite
while read -r hostname; do
#Tulosta nrpe.cfg-tiedostoon server_address=oma_ip-osoite (aiempi
server_address kommentoituna, ei vaikutusta)
printf '%s\n' "server_address=$hostname" >> /etc/nagios/nrpe.cfg
done < <(hostname -i)

#Step 3: nrpe.cfg muokkaus

#Poistetaan allowed_hosts-tieto
sed -i -e "/allowed_hosts=127.0.0.1/d" /etc/nagios/nrpe.cfg
#Tulosta tiedoston loppuun Nagios-palvelimen IP allowed_hosts-tietoon
printf "allowed_hosts=127.0.0.1,$nagiosip\n" >> /etc/nagios/nrpe.cfg

#Tulosta tiedoston loppuun check_ping-komento kohteena aiemmin
valittu IP-osoite
printf '%s\n' "command[check_ping]=/usr/lib/nagios/plugins/check_ping
-H $kohde -w 100,10% -c 250,50%" >> /etc/nagios/nrpe.cfg

# Step 3: palvelun bootti

service nagios-nrpe-server restart

# Step 4: Ilmoitus asennuksen päättymisestä

printf "\nAsennus tehty, jatka konfigurointia Nagios-palve-
limella\n\n."
```

Liite 5. Nagios-palvelimen skripti

```
#! /bin/bash
#Skripti Nagioksen .cfg tiedostojen tekoon (template: linux-
server).
#Lisää palveluista myös levytilan tarkastamisen ja ping-
komennon.
#Lopuksi käynnistää Nagios-prosessin uudestaan

#Step 1: Kerätään tiedot
printf "\nNagioksessa käytettävä palvelimen nimitys/alias? (Ei
välejä tai erikoismerkkejä)\n"

read palvelimen_nimi

printf "\nPalvelimen IP-osoite? (esim. 192.168.0.1)\n"

read ip_osoite

#Step 2: Lisätään .cfg-tiedosto Nagioksen konfiguraatioihin
vaiheessa 1 annetuilla tiedoilla
printf "define host {
        use                     linux-server
        host_name               $palvelimen_nimi
        alias                   $palvelimen_nimi
        address                 $ip_osoite
}
define service{
        use                             generic-service
        host_name                       $palvelimen_nimi
        service_description             HDD
        check_command                   check_nrpe!check_hda1
}
define service{
        use                             generic-service
        host_name                       $palvelimen_nimi
        service_description             PING
        check_command                   check_nrpe!check_ping
}

" >> /usr/local/nagios/etc/servers/$palvelimen_nimi.cfg

#Step 3: Käynnistetään Nagios-prosessi uudestaan

printf "\nPalvelin $palvelimen_nimi lisätty valvottaviin
kohteisiin.\n\n"

service nagios restart
```